Part 2

Building virtual infrastructure consisting of computers and networking

Computing power and network connectivity has become a basic need for private households, medium-sized enterprises, and big corporations. Operating hardware in data centers that are in-house or outsourced has covered these needs in the past. Now the cloud is revolutionizing the way you can access computing power.

Virtual machines can be started and stopped on-demand to fulfill your computing needs within minutes. Being able to install software on virtual machines enables you to execute your computing tasks without needing to buy or rent hardware.

If you want to understand AWS, you have to dive into the possibilities of the API working behind the scenes. You can control every single service on AWS by sending requests to a REST API. Based on this, there is a variety of solutions that help you to automate your overall infrastructure. Infrastructure automation is a big advantage of the cloud compared to hosting on-premises. This chapter will guide you into infrastructure orchestration and the automated deployment of applications.

Creating *virtual networks* allows you to build closed and secure network environments on AWS and to connect these networks with your home or corporate network.

Chapter 3 covers working with virtual machines. You will learn about the key concepts of the EC2 service.

Chapter 4 contains different approaches to automate your infrastructure. You will learn how to make use of Infrastructure-as-Code.

Chapter 5 shows three ways of deploying your software to AWS.

Chapter 6 is about networking. You will learn how to secure your system with a virtual private network and firewalls.

Chapter 7 is about a new way of computing: functions. You will learn how to automate operational tasks with AWS Lambda.

3 Using virtual machines: EC2

This chapter covers

- Launching a virtual machine with Linux
- Controlling a virtual machine remotely via SSH
- Monitoring and debugging a virtual machine
- Saving costs for virtual machines

It's impressive what you can achieve with the computing power of the smartphone in your pocket or the laptop in your bag. But if your task requires massive computing power or high network traffic, or needs to run reliably 24/7, a virtual machine is a better fit. With a virtual machine, you get access to a slice of a physical machine located in a data center. On AWS, virtual machines are offered by the service called Elastic Compute Cloud (EC2).

Not all examples are covered by Free Tier

The examples in this chapter are *not* all covered by the Free Tier. A special warning message appears when an example incurs costs. As for the other examples, as long as you don't run them longer than a few days, you won't pay anything for them. Keep in mind that this applies only if you created a fresh AWS account for this book and nothing else is going on in your AWS account. Try to complete the chapter within a few days; you'll clean up your account at the end.

3.1 Exploring a virtual machine

A virtual machine (VM) is a part of a physical machine that's isolated by software from other VMs on the same physical machine; it consists of CPUs, memory, networking interfaces, and storage. The physical machine is called the *host machine*, and the VMs running on it are called *guests*. A *hypervisor* is responsible for isolating the guests from each other and for scheduling requests to the hardware, by providing a virtual hardware platform to the guest system. Figure 3.1 shows these layers of virtualization.

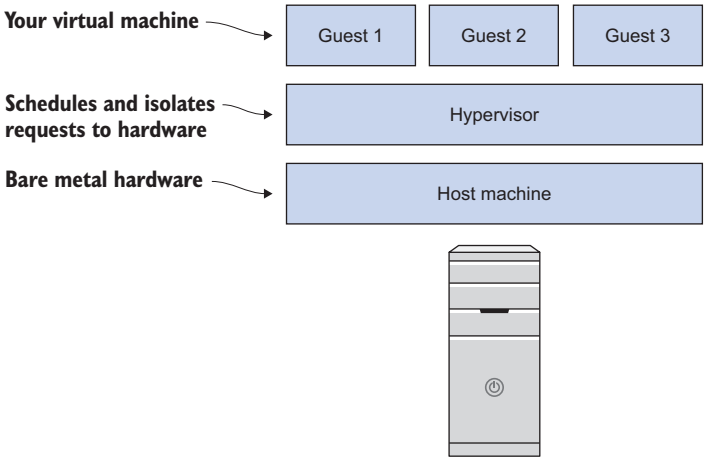


Figure 3.1 Layers of virtualization

Typical use cases for a virtual machine are as follows:

- Hosting a web application such as WordPress
- Operating an enterprise application, such as an ERP application
- Transforming or analyzing data, such as encoding video files

3.1.1 Launching a virtual machine

It takes only a few clicks to launch a virtual machine:

1. Open the AWS Management Console at https://console.aws.amazon.com.
2. Make sure you're in the N. Virgina (US East) region (see figure 3.2), because we optimized our examples for this region.
3. Find the EC2 service in the navigation bar under Services, and click it. You'll see a page like the one in figure 3.3.
4. Click Launch Instance to start the wizard for launching a virtual machine.

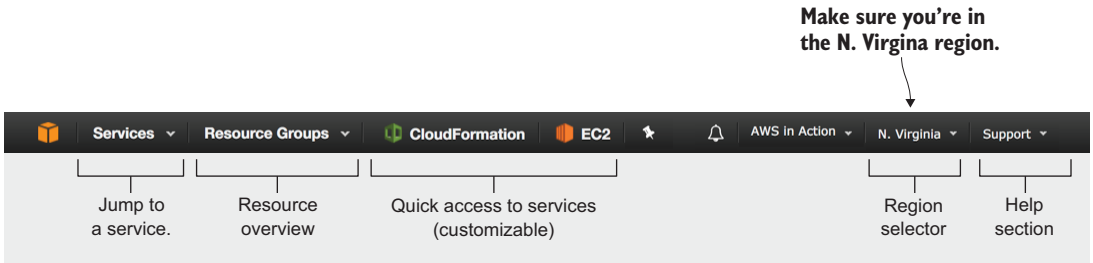


Figure 3.2 Examples are optimized for the N. Virginia region

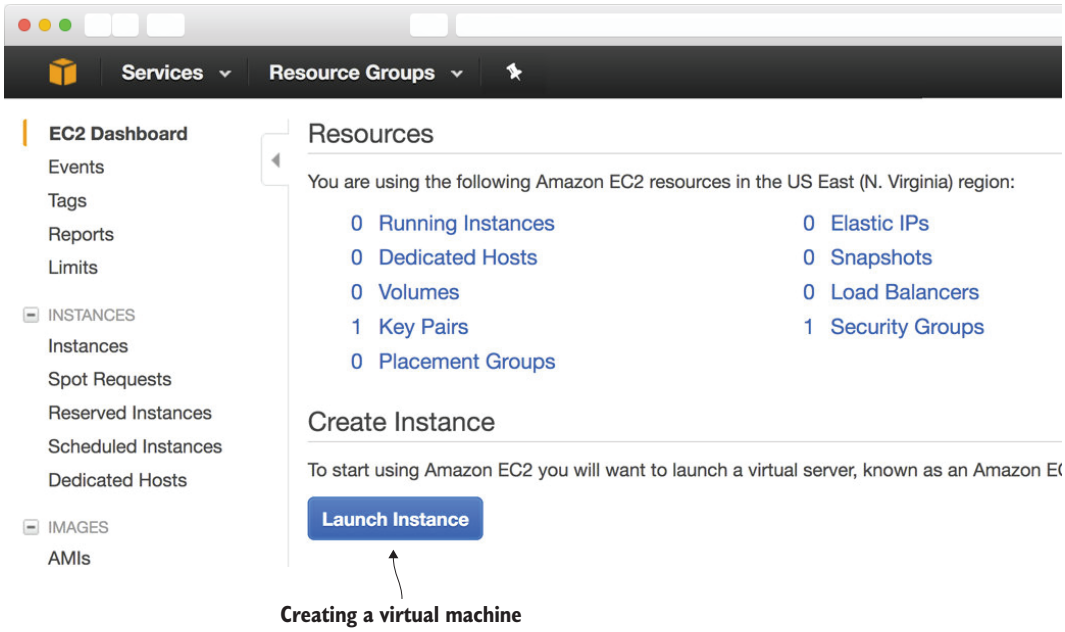


Figure 3.3 The EC2 Dashboard gives you an overview of all parts of the service.

The wizard in figure 3.3 will guide you through the following steps:

1. Selecting an OS
2. Choosing the size of your virtual machine
3. Configuring details
4. Adding storage
5. Tagging your virtual machine
6. Configuring a firewall
7. Reviewing your input and selecting a key pair used for SSH authentication

SELECTING THE OPERATING SYSTEM

The first step is to choose an OS. In AWS, the OS comes bundled with preinstalled software for your virtual machine; this bundle is called an *Amazon Machine Image (AMI)*. Select Ubuntu Server 16.04 LTS (HVM), as shown in figure 3.4. Why Ubuntu? Because Ubuntu offers a ready-to-install package named *linkchecker* that you'll use later to check a website for broken links.

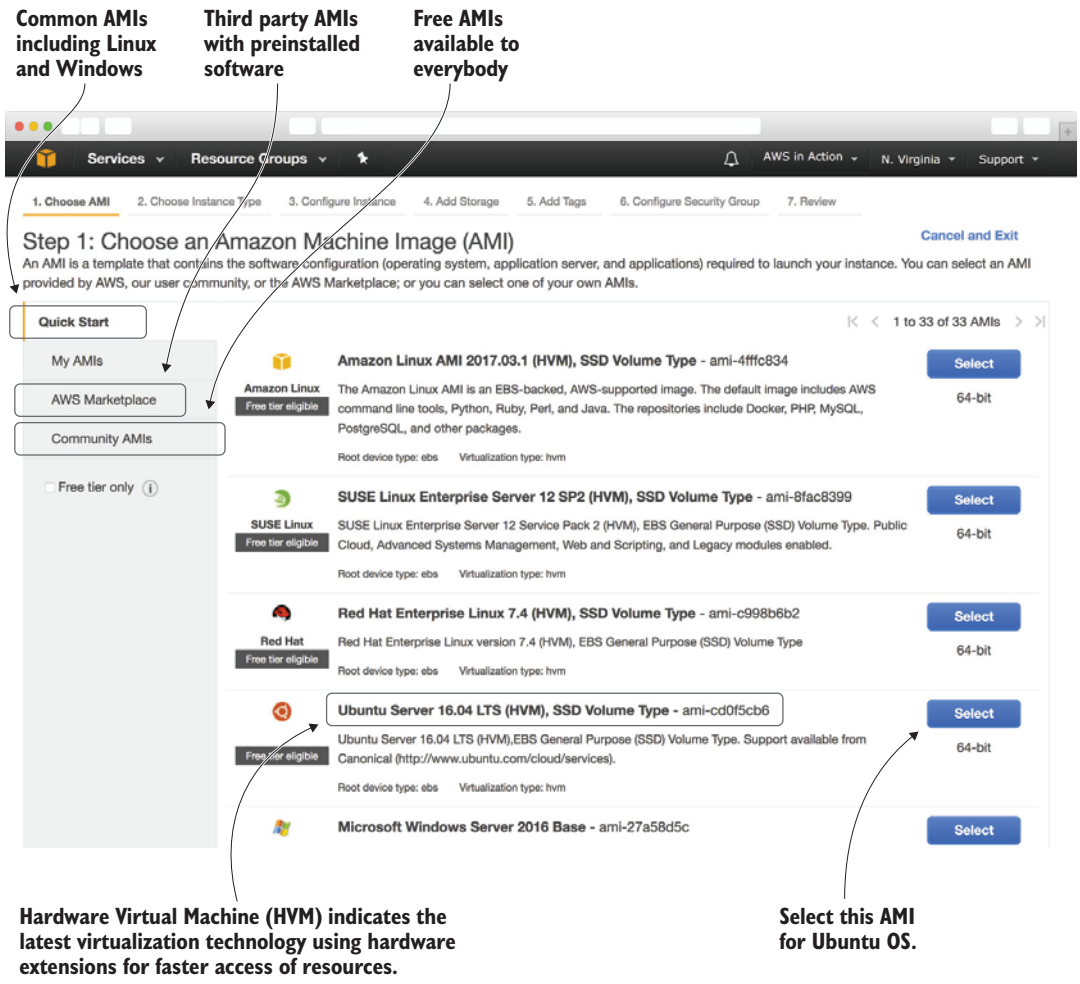


Figure 3.4 Choose the OS for your virtual machine.

The AMI is the basis your virtual machine starts from. AMIs are offered by AWS, third-party providers, and by the community. AWS offers the Amazon Linux AMI, which is based on Red Hat Enterprise Linux and optimized for use with EC2. You'll also find popular Linux distributions and AMIs with Microsoft Windows Server, and you can find more AMIs with preinstalled third-party software in the AWS Marketplace.

Amazon Linux 2 is coming

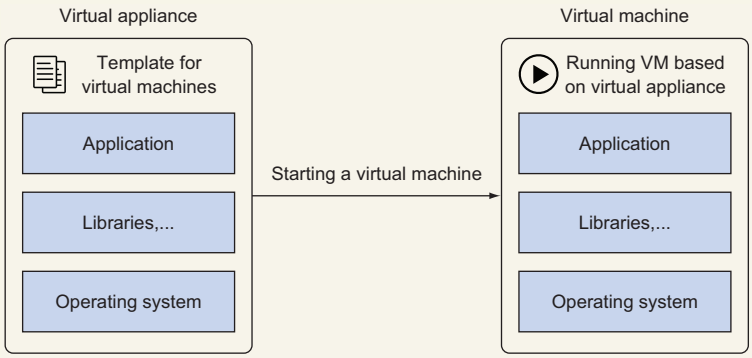
Amazon Linux 2 is the next generation Amazon Linux OS. At the time of writing, no LTS release of Amazon Linux 2 was available. But now, as you read this book, that might have changed. Check out the Amazon Linux 2 FAQs: https://aws.amazon.com/amazon-linux-2/faqs/.

Amazon Linux 2 adds long-term support for five years. You can now run Amazon Linux 2 locally and on-premises as well. Besides that, systemd is now used and a new mechanism called the *extras library* provides up-to-date versions of software bundles such as NGINX.

When choosing an AMI, start by thinking about the requirements of the application you want to run on the VM. Your knowledge and experience with a specific operating system is another important factor when deciding which AMI to start with. It's also important that you trust the AMI's publisher. We prefer working with Amazon Linux, as it is maintained and optimized by AWS.

Virtual appliances on AWS

A *virtual appliance* is an image of a virtual machine containing an OS and preconfigured software. Virtual appliances are used when the hypervisor starts a new VM. Because a virtual appliance contains a fixed state, every time you start a VM based on a virtual appliance, you'll get exactly the same result. You can reproduce virtual appliances as often as needed, so you can use them to eliminate the cost of installing and configuring complex stacks of software. Virtual appliances are used by virtualization tools from VMware, Microsoft, and Oracle, and for infrastructure-as-a-service (IaaS) offerings in the cloud.



A virtual appliance contains a template for a virtual machine

The AMI is a special type of virtual appliance for use with the EC2 service. An AMI technically consists of a read-only filesystem including the OS, additional software, and configuration; it doesn't include the kernel of the OS. The kernel is loaded from an Amazon Kernel Image (AKI). You can also use AMIs for deploying software on AWS.

(continued)

AWS uses Xen, an open source hypervisor. The current generations of VMs on AWS use hardware-assisted virtualization. The technology is called Hardware Virtual Machine (HVM). A virtual machine run by an AMI based on HVM uses a fully virtualized set of hardware, and can take advantage of extensions that provide fast access to the underlying hardware.

Using a version 3.8+ kernel for your Linux-based VMs will provide the best performance. To do so, you should use at least Amazon Linux 13.09, Ubuntu 14.04, or RHEL7. If you're starting new VMs, make sure you're using HVM images.

In November 2017 AWS announced a new generation of virtualization called Nitro. Nitro combines a KVM-based hypervisor with customized hardware (ASICs) aiming to provide a performance that is indistinguishable from bare metal machines. Currently, the c5 and m5 instance types make use of Nitro, and it's likely that more instance families using Nitro will follow.

CHOOSING THE SIZE OF YOUR VIRTUAL MACHINE

It's now time to choose the computing power needed for your virtual machine. Figure 3.5 shows the next step of the wizard. On AWS, computing power is classified into instance types. An instance type primarily describes the number of virtual CPUs and the amount of memory.

Table 3.1 shows examples of instance types for different use cases. The prices represent the actual prices in USD for a Linux VM in the US East (N. Virginia) region, as recorded Aug. 31, 2017.

Table 3.1 Examples of instance families and instance types

Instance type	Virtual CPUs	Memory	Description	Typical use case	Hourly cost (USD)
t2.nano	1	0.5 GB	Smallest and cheapest instance type, with moderate baseline performance and the ability to burst CPU performance above the baseline	Testing and development environments, and applications with very low traffic	0.0059
m4.large	2	8 GB	Has a balanced ratio of CPU, memory, and networking performance	All kinds of applications, such as medium databases, web servers, and enterprise applications	0.1
r4.large	2	15.25 GB	Optimized for memory-intensive applications with extra memory	In-memory caches and enterprise application servers	0.133

There are instance families optimized for different kinds of use cases.

- *T family*—Cheap, moderate baseline performance with the ability to burst to higher performance for short periods of time
- *M family*—General purpose, with a balanced ration of CPU and memory

- *C family*—Computing optimized, high CPU performance
- *R family*—Memory optimized, with more memory than CPU power compared to M family
- *D family*—Storage optimized, offering huge HDD capacity
- *I family*—Storage optimized, offering huge SSD capacity
- *X family*—Extensive capacity with a focus on memory, up to 1952 GB memory and 128 virtual cores
- *F family*—Accelerated computing based on FPGAs (field programmable gate arrays)
- *P, G, and CG family*—Accelerated computing based on GPUs (graphics processing units)

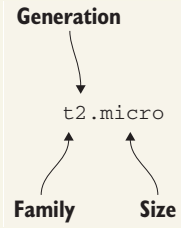
Our experience indicates that you'll overestimate the resource requirements for your applications. So, we recommend that you try to start your application with a smaller instance type than you think you need at first. You can change the instance family and type later if needed.

Instance types and families

The names for different instance types are all structured in the same way. The *instance family* groups instance types with similar characteristics. AWS releases new instance types and families from time to time; the different versions are called *generations*. The *instance size* defines the capacity of CPU, memory, storage, and networking.

The instance type *t2.micro* tells you the following:

- The instance family is called *t*. It groups small, cheap virtual machines with low baseline CPU performance but the ability to burst significantly over baseline CPU performance for a short time.
- You're using generation *2* of this instance family.
- The size is *micro*, indicating that the EC2 instance is very small.



Disassembling the t2.micro instance type

Computer hardware is getting faster and more specialized, so AWS is constantly introducing new instance types and families. Some of them are improvements of existing instance families, and others are focused on specific workloads. For example, the instance family R4 was introduced in November 2016. It provides instances for memory-intensive workloads and improves the R3 instance types.

One of the smallest and cheapest VMs will be enough for your first experiments. In the wizard screen shown in figure 3.5, choose the instance type t2.micro, which is eligible for the Free Tier. Then click Next: Configure Instance Details to proceed.

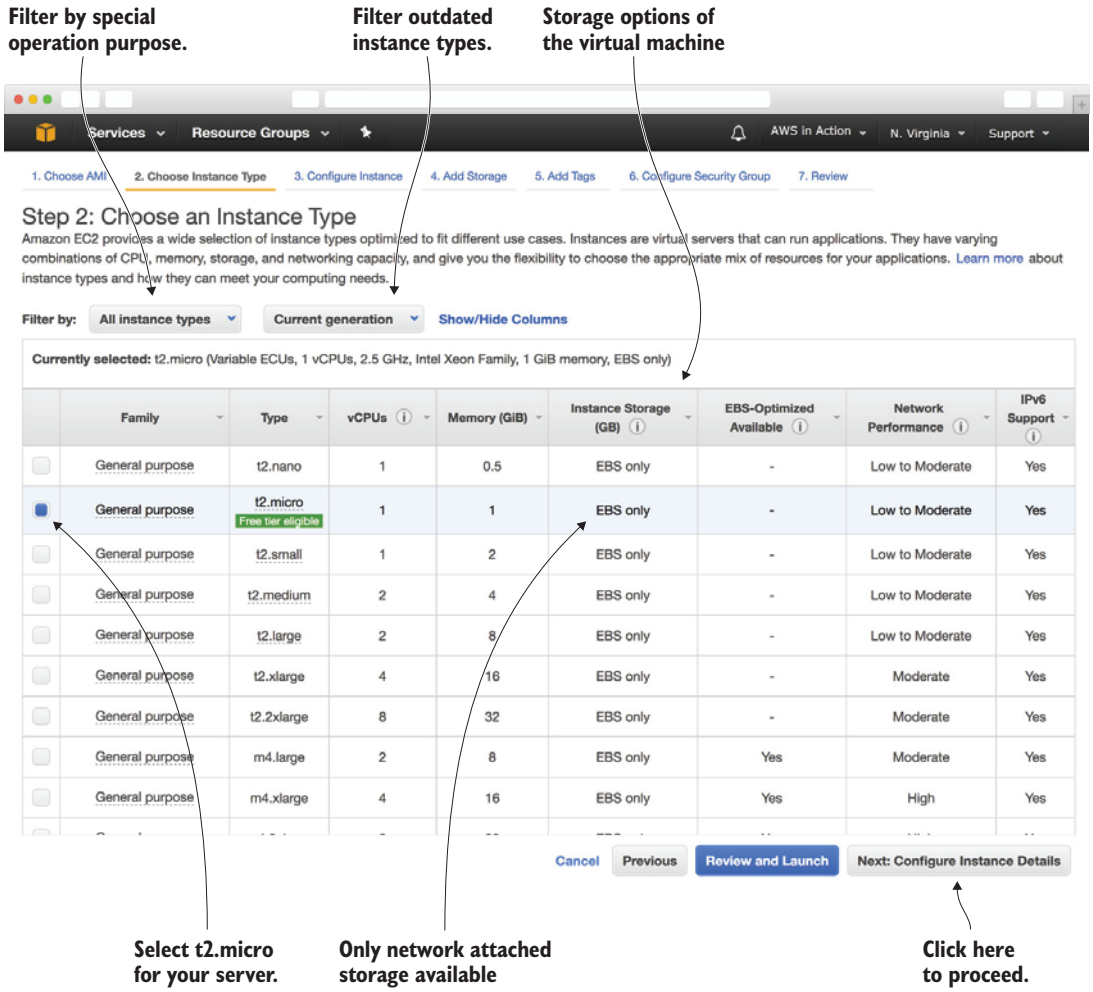


Figure 3.5 Choosing the size of your virtual machine

CONFIGURING DETAILS, STORAGE, FIREWALL, AND TAGS

The next four steps of the wizard (shown in figure 3.6–figure 3.9) are easy, because you don't need to change the defaults. You'll learn about these settings in detail later in the book.

Figure 3.6 shows where you can change the details for your VM, such as the network configuration or the number of VMs to launch. For now, keep the defaults, and click Next: Add Storage to proceed.

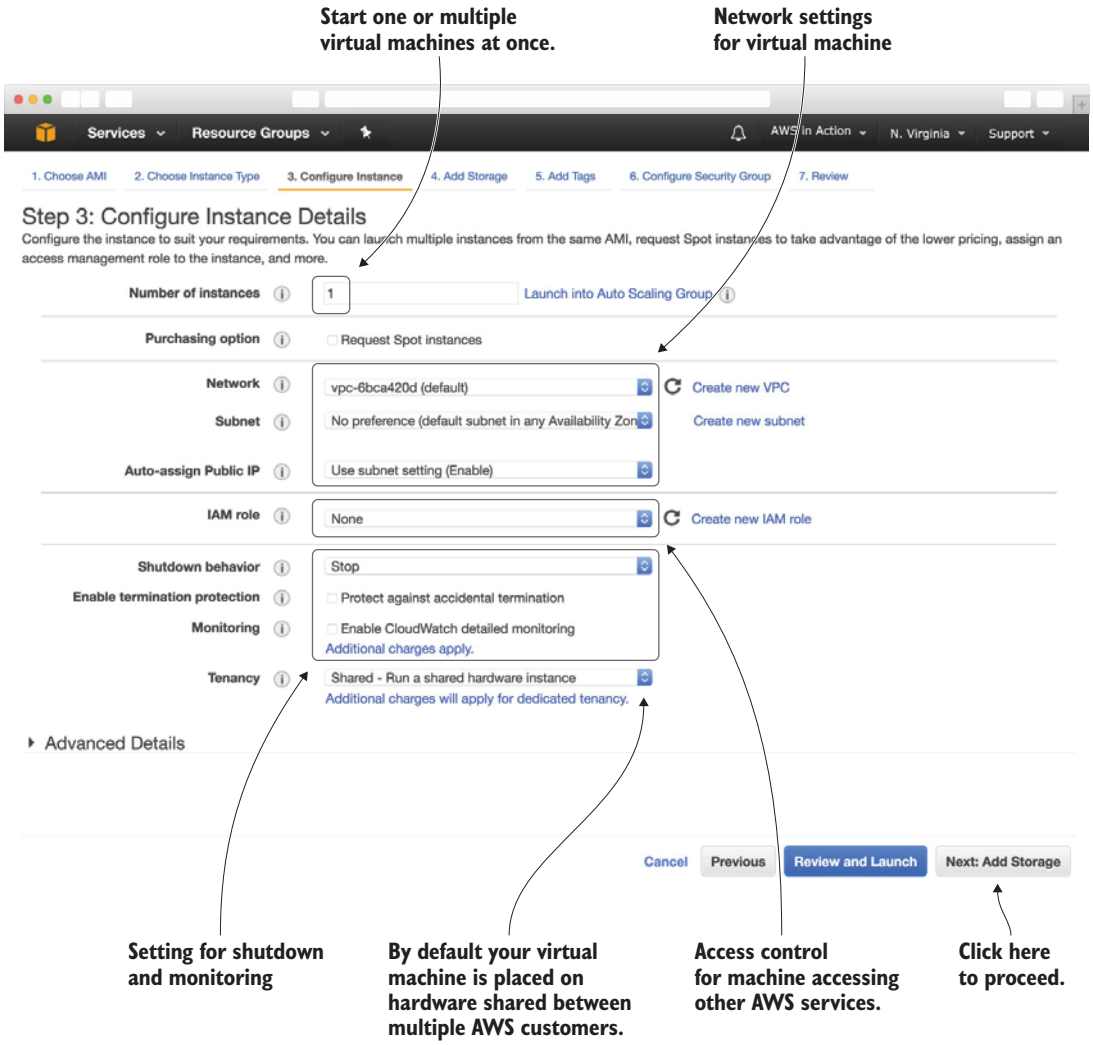


Figure 3.6 Details for the virtual machine

There are different options for storing data on AWS, which we'll cover in detail in the following chapters. Figure 3.7 shows the screen where you can add network-attached storage to your virtual machine. Keep the defaults, and click Next: Add Tags.

A tidy house indicates a tidy mind. Tags help you organize resources on AWS. Figure 3.8 shows the Add Tags screen. Each tag is nothing more than a key-value pair. For this example, add at least a `Name` tag to help you find your stuff later. Use `Name` as the key and `mymachine` as the value, as shown in the figure. Then click Next: Configure Security Group.

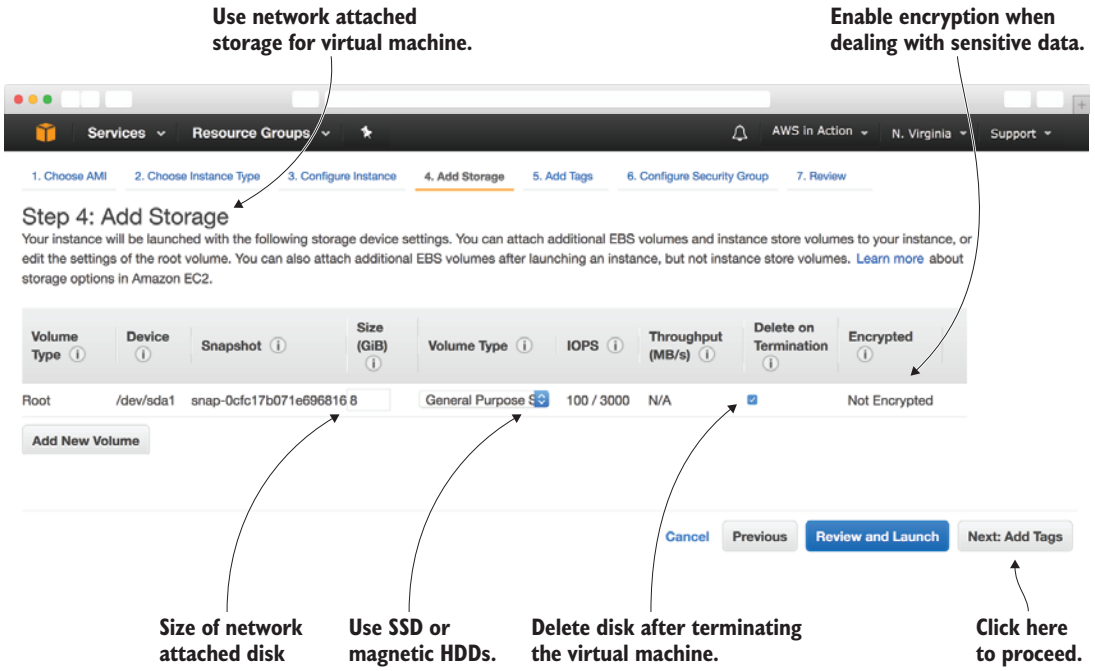


Figure 3.7 Adding network-attached storage to your virtual machine

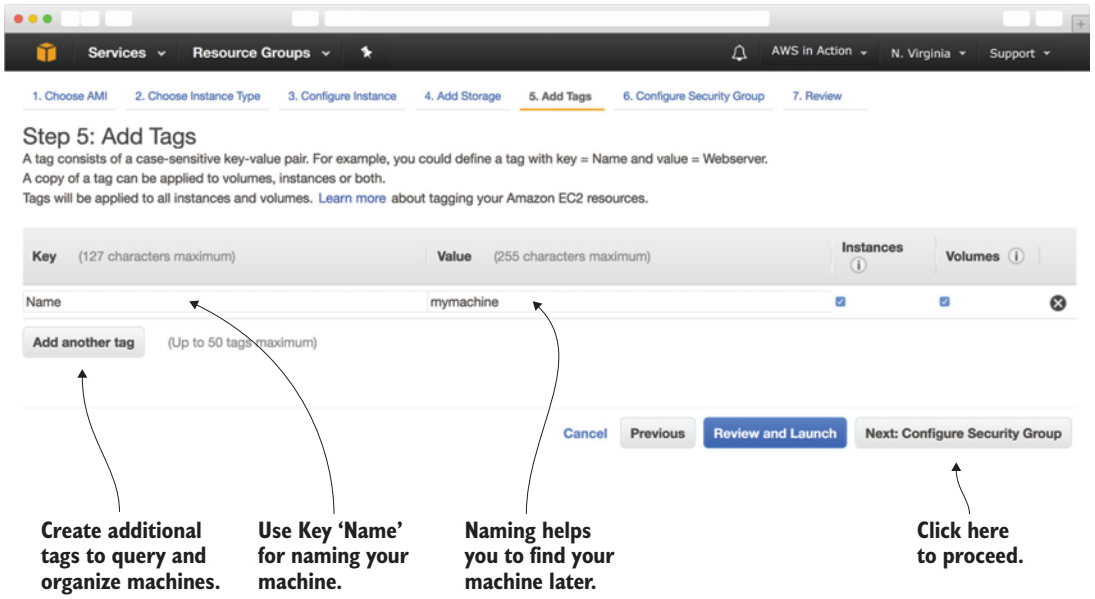


Figure 3.8 Tagging your virtual machine with a `Name` tag

Organizing AWS resources with tags

Most AWS resources can be tagged. For example, you can add tags to an EC2 instance. There are three major use cases for resource tagging:

1. Use tags to filter and search for resources.
2. Analyze your AWS bill based on resource tags.
3. Restrict access to resources based on tags.

Typical tags include environment type (such as test or production), responsible team, department, and cost center.

A firewall helps to secure your virtual machine. Figure 3.9 shows the settings for a default firewall allowing access via SSH from anywhere.

1. Select Create a new security group.
2. Type in `ssh-only` for the name and description of the security group.
3. Keep the default rule allowing SSH from anywhere.
4. Click Review and Launch to proceed with the next step.

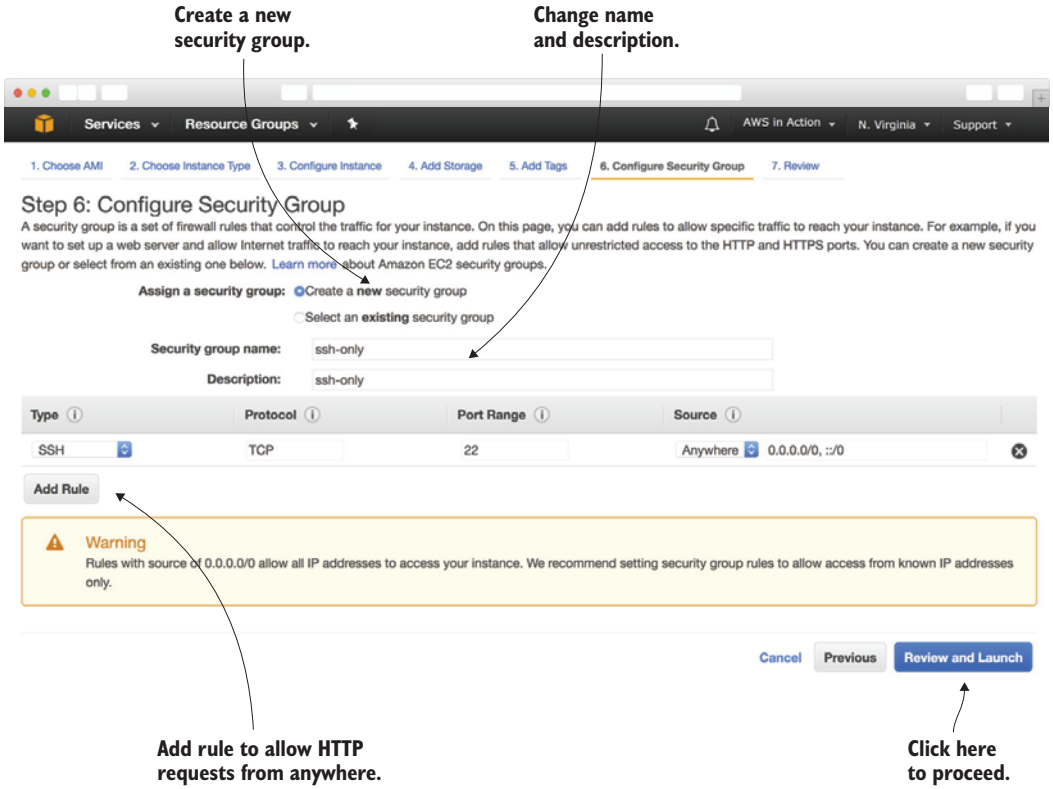


Figure 3.9 Configuring the firewall for your virtual machine

REVIEWING YOUR INPUT AND SELECTING A KEY PAIR FOR SSH

You're almost finished. The wizard should show a review of your new virtual machine (see figure 3.10). Make sure you chose Ubuntu Server 16.04 LTS (HVM) as the OS and t2.micro as the instance type. If everything is fine, click the Launch button. If not, go back and make changes to your VM where needed.

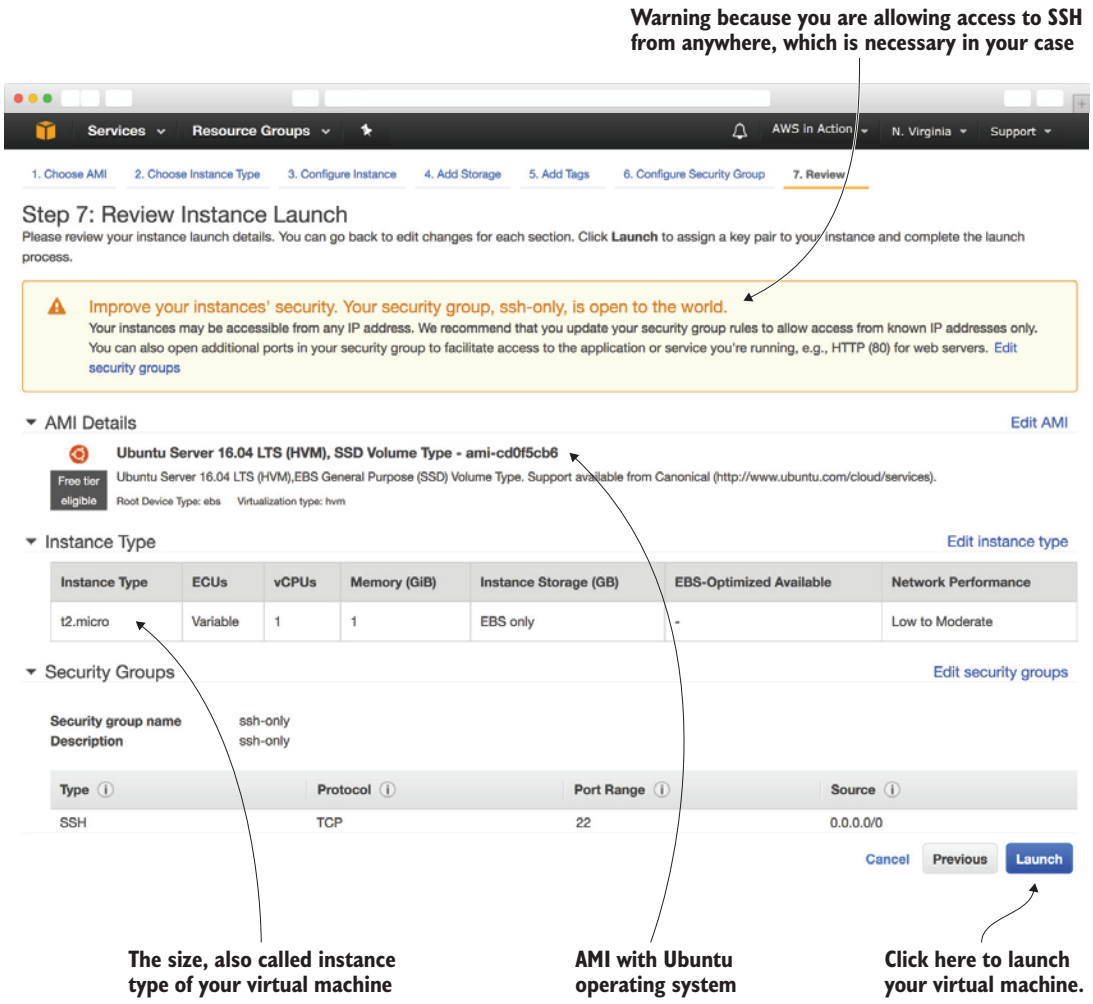


Figure 3.10 Review launch for the virtual machine

Last but not least, the wizard asks for your new virtual machine's key. Choose the option Choose an Existing Key Pair, select the key pair `mykey`, and click Launch Instances (see figure 3.11).

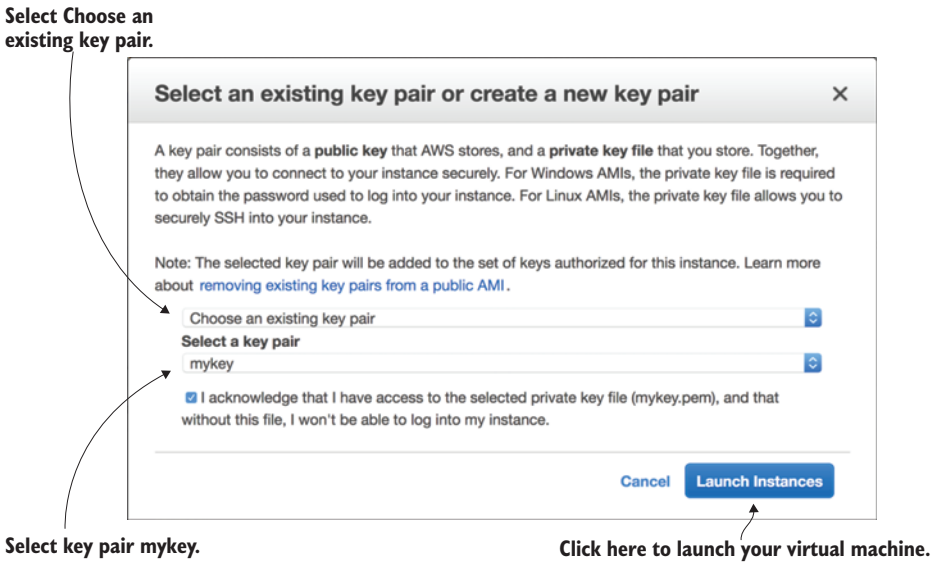


Figure 3.11 Choosing a key pair for your virtual machine

Missing your key?

Logging in to your virtual machine requires a key. You use a key instead of a password to authenticate yourself. Keys are much more secure than passwords, and using keys for SSH is enforced for VMs running Linux on AWS. If you skipped the creation of a key in section 1.8.3, follow these steps to create a personal key:

1. Open the AWS Management Console at https://console.aws.amazon.com. Find the EC2 service in the navigation bar under Services, and click it.
2. Switch to Key Pairs via the submenu.
3. Click Create Key Pair.
4. Enter `mykey` for Key Pair Name, and click Create. Your browser downloads the key automatically.
5. Open your terminal, and switch to your download folder.

6a. Linux and macOS only: change the access rights of the file mykey.pem by running `chmod 400 mykey.pem` in your terminal.

6b. Windows only: Windows doesn't ship an SSH client yet, so you need to install PuTTY. PuTTY comes with a tool called PuTTYgen that can convert the mykey.pem file into a mykey.ppk file, which you'll need. Open PuTTYgen, and select SSH-2 RSA under Type of Key to Generate. Click Load. Because PuTTYgen displays only *.ppk files, you need to switch the file extension of the File Name Input to All Files. Now you can select the mykey.pem file and click Open. Click OK in the confirmation dialog box. Change Key Comment to `mykey`. Click Save Private Key. Ignore the warning about saving the key without a passphrase. Your .pem file is now converted to the .ppk format needed by PuTTY.

You'll find a more detailed explanation about how to create a key in chapter 1.

Your virtual machine should now launch. Open an overview by clicking View Instances, and wait until the machine reaches the Running state. To take full control over your virtual machine, you need to log in remotely.

3.1.2 Connecting to your virtual machine

Installing additional software and running commands on your virtual machine can be done remotely. To log in to the virtual machine, you have to figure out its public domain name:

1. Click the EC2 service in the navigation bar under Services, and click Instances in the submenu at left to jump to an overview of your virtual machine.

   Select the virtual machine from the table by clicking it. Figure 3.12 shows the overview of your virtual machines and the available actions.

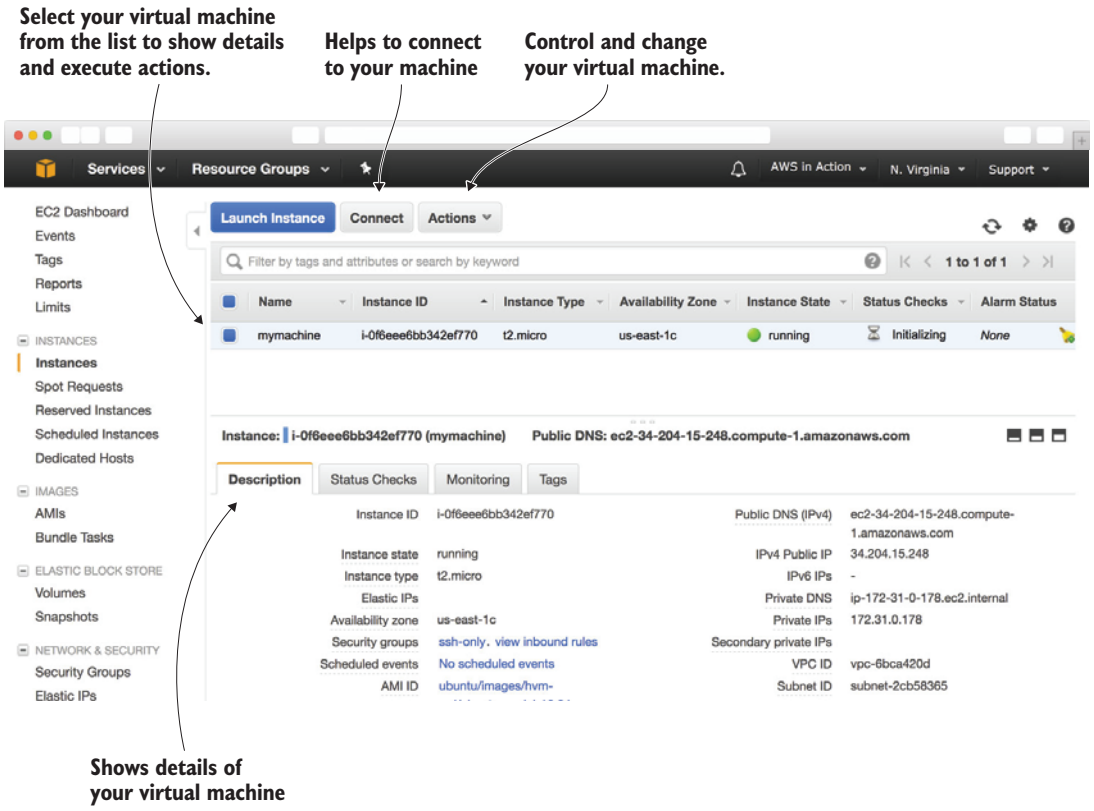


Figure 3.12 Overview of your virtual machines with actions to control them

2. Click Connect to open the instructions for connecting to the virtual machine.

   Figure 3.13 shows the dialog with instructions for connecting to the virtual machine. Find the public DNS of your virtual machine, such as `ec2-34-204-15-248.compute-1.amazonaws.com` in our example.

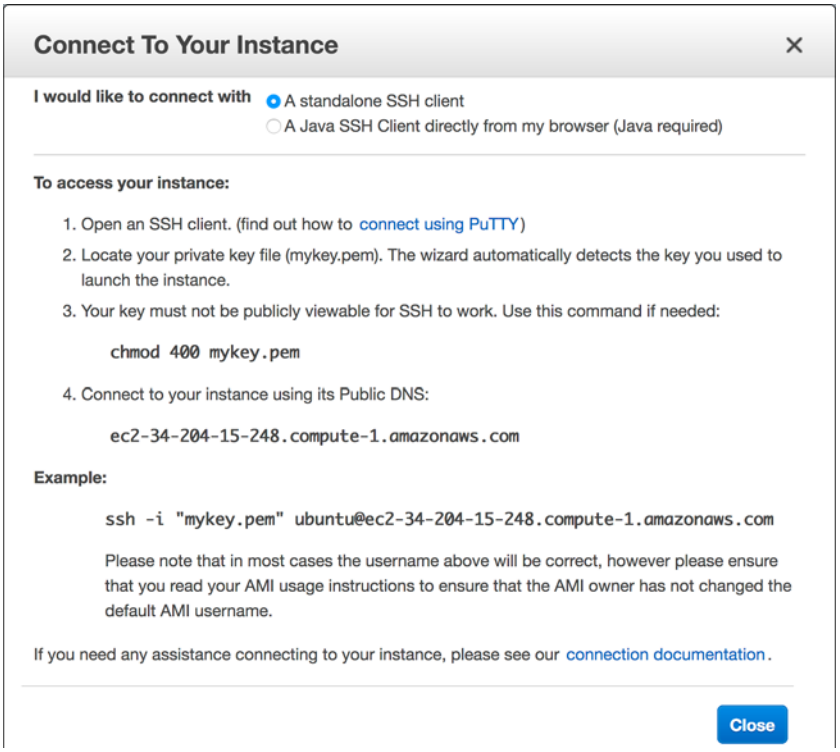


Figure 3.13 Instructions for connecting to the virtual machine with SSH

With the public DNS and your key, you can connect to your virtual machine. Continue to the next section, depending on your OS.

Linux and macOS

Open your terminal, and type `ssh -i $PathToKey/mykey.pem ubuntu@$PublicDns`, replacing *`$PathToKey`* with the path to the key file you downloaded in section 1.8.3 and *`$PublicDns`* with the public DNS shown in the Connect dialog in the AWS Management Console. You'll see a security alert regarding the authenticity of the new host. Answer yes to connect.

Windows

For Windows, follow these steps:

1. Find the mykey.ppk file you created in section 1.8.3, and double-click to open it.
2. PuTTY Pageant should appear in the Windows taskbar as an icon. If not, you may need to install or reinstall PuTTY as described in section 1.8.3.
3. Start PuTTY. Fill in the public DNS shown in the Connect dialog in the AWS Management Console, and click Open (see figure 3.14).
4. You'll see a security alert regarding the authenticity of the new host. Answer yes and type `ubuntu` as the login name. Click Enter.

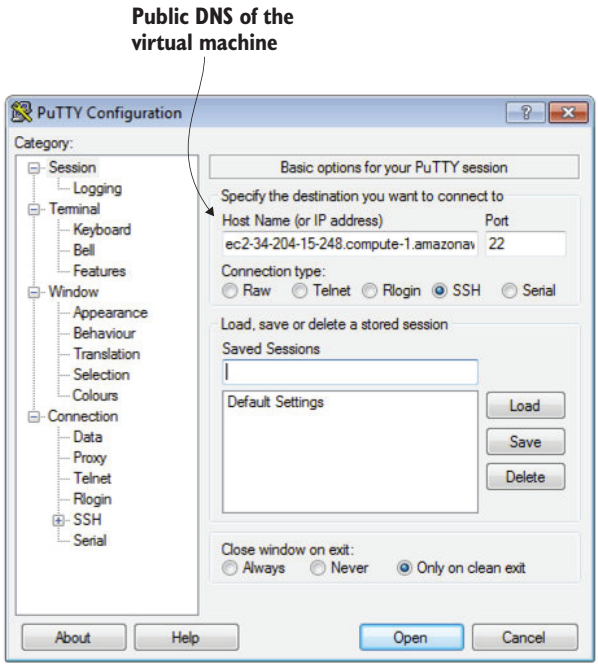


Figure 3.14 Connecting to the virtual machine with PuTTY on Windows

LOGIN MESSAGE

Whether you're using Linux, Mac OS, or Windows, after a successful login you should see a message like the following:

```
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-1022-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

0 packages can be updated.
0 updates are security updates.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-0-178:~$
```

You're now connected to your virtual machine and are ready to run a few commands.

3.1.3 Installing and running software manually

You've now started a virtual machine with an Ubuntu OS. It's easy to install additional software with the help of the package manager apt. First we need to make sure the package manager is up-to-date. Please run the following command to update the list of available packages:

```
$ sudo apt-get update
```

To begin, you'll install a tiny tool called `linkchecker` that allows you to find broken links on a website:

```
$ sudo apt-get install linkchecker -y
```

Now you're ready to check for links pointing to websites that no longer exist. To do so, choose a website and run the following command:

```
$ linkchecker https://...
```

The output of checking the links looks something like this:

```
[...]
URL        `http://www.linux-mag.com/blogs/fableson'
Name       `Frank Ableson's Blog'
Parent URL http://manning.com/about/blogs.html, line 92, col 27
Real URL   http://www.linux-mag.com/blogs/fableson
Check time 1.327 seconds
Modified   2015-07-22 09:49:39.000000Z
Result     Error: 404 Not Found

URL        `/catalog/dotnet'
Name       `Microsoft & .NET'
Parent URL http://manning.com/wittig/, line 29, col 2
Real URL   http://manning.com/catalog/dotnet/
Check time 0.163 seconds
D/L time   0.146 seconds
Size       37.55KB
Info       Redirected to `http://manning.com/catalog/dotnet/'.
           235 URLs parsed.
Modified   2015-07-22 01:16:35.000000Z
Warning    [http-moved-permanent] HTTP 301 (moved permanent)
           encountered: you should update this link.
Result     Valid: 200 OK
[...]
```

Depending on the number of web pages, the crawler may need some time to check all of them for broken links. At the end, it lists the broken links and gives you the chance to find and fix them.

3.2 Monitoring and debugging a virtual machine

If you need to find the reason for an error or why your application isn't behaving as you expect, it's important to have access to tools that can help with monitoring and debugging. AWS provides tools that let you monitor and debug your virtual machines. One approach is to examine the virtual machine's logs.

3.2.1 Showing logs from a virtual machine

If you need to find out what your virtual machine was doing during and after startup, there is a simple solution. AWS allows you to see the EC2 instance's logs with the help of the Management Console (the web interface you use to start and stop virtual machines). Follow these steps to open your VM's logs:

1. Open the EC2 service from the main navigation, and select Instances from the submenu.
2. Select the running virtual machine by clicking the row in the table.
3. In the Actions menu, choose Instance Settings > Get System Log.

A window opens and shows you the system logs from your VM that would normally be displayed on a physical monitor during startup (see figure 3.15).

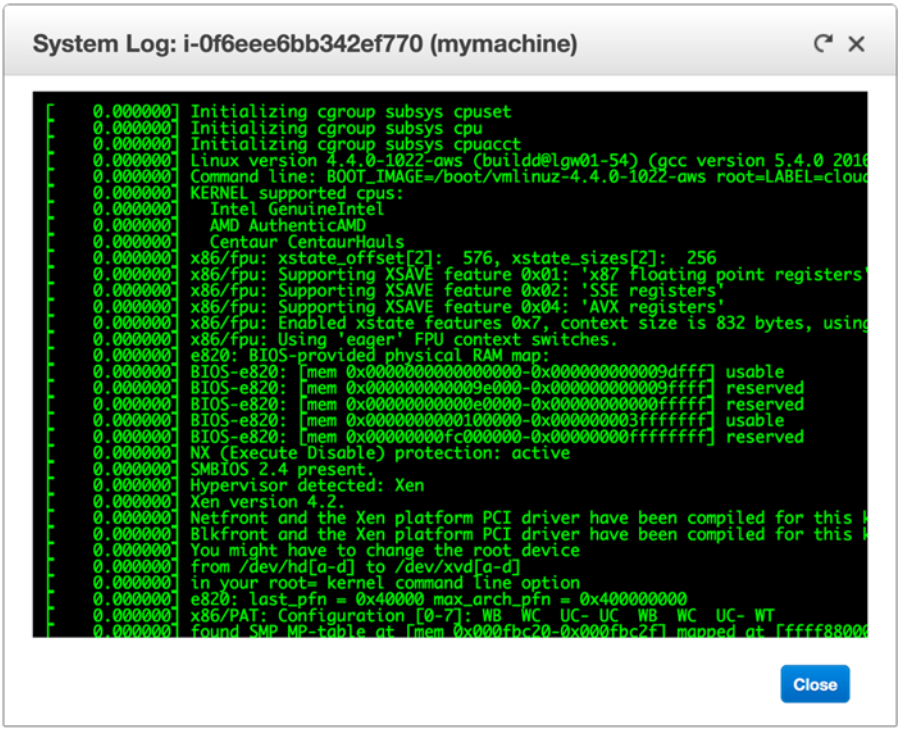


Figure 3.15 Debugging a virtual machine with the help of logs

The log contains all log messages that would be displayed on the monitor of your machine if you were running it on-premises. Watch out for any log messages stating that an error occurred during startup. If the error message is not obvious, you should contact the vendor of the AMI, AWS Support, or post your question in the AWS Developer Forums at https://forums.aws.amazon.com.

This is a simple and efficient way to access your system logs without needing an SSH connection. Note that it will take several minutes for a log message to appear in the log viewer.

3.2.2 Monitoring the load of a virtual machine

AWS can help you answer another question: is your virtual machine close to its maximum capacity? Follow these steps to open the EC2 instance's metrics:

1. Open the EC2 service from the main navigation, and select Instances from the submenu.
2. Select the running virtual machine by clicking the appropriate row in the table.
3. Select the Monitoring tab at lower right.
4. Click the Network In chart to dive into the details.

You'll see a graph that shows the virtual machine's utilization of incoming networking traffic, similar to figure 3.16. There are metrics for CPU, network, and disk usage. As AWS is looking at your VM from the outside, there is no metric indicating the memory usage. You can publish a memory metric yourself, if needed. The metrics are updated every 5 minutes if you use basic monitoring, or every minute if you enable detailed monitoring of your virtual machine, which costs extra.

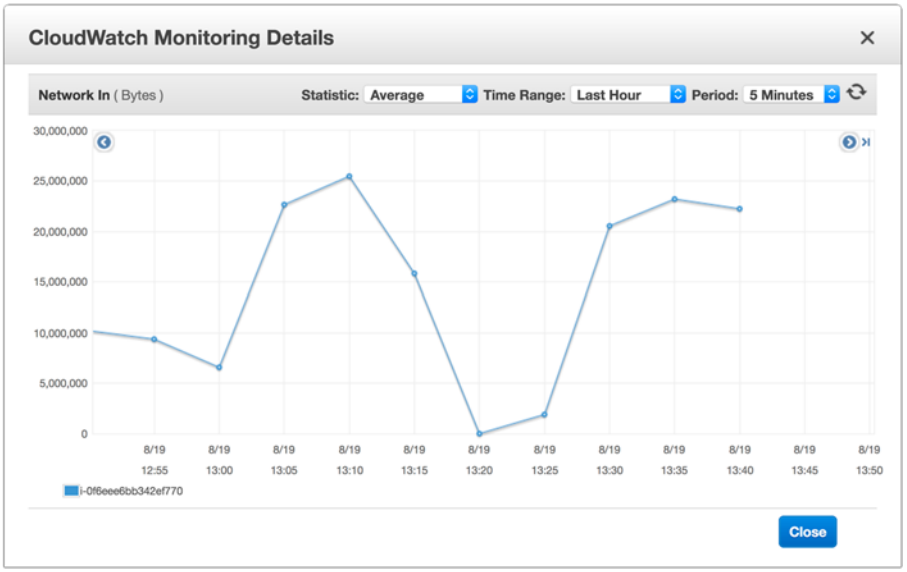


Figure 3.16 Gaining insight into a virtual machine's incoming network traffic with the CloudWatch metric

Checking the metrics of your EC2 instance is helpful when debugging performance issues. You will also learn how to increase or decrease your infrastructure based on these metrics in chapter 17.

Metrics and logs help you monitor and debug your virtual machines. Both tools can help ensure that you're providing high-quality services in a cost-efficient manner. Look at Monitoring Amazon EC2 in the AWS documentation at http://mng.bz/0q40 if you are looking for more detailed information about monitoring your virtual machines.

3.3 Shutting down a virtual machine

To avoid incurring charges, you should always turn off virtual machines you're not using them. You can use the following four actions to control a virtual machine's state:

- *Start*—You can always start a stopped virtual machine. If you want to create a completely new machine, you'll need to launch a virtual machine.
- *Stop*—You can always stop a running virtual machine. A stopped virtual machine doesn't incur charges, expect for attached resources like network-attached storage. A stopped virtual machine can be started again but likely on a different host. If you're using network-attached storage, your data persists.
- *Reboot*—Have you tried turning it off and on again? If you need to reboot your virtual machine, this action is what you want. You won't lose any persistent data when rebooting a virtual machine because it stays on the same host.
- *Terminate*—Terminating a virtual machine means deleting it. You can't start a virtual machine that you've already terminated. The virtual machine is deleted, usually together with dependencies like network-attached storage and public and private IP addresses. A terminated virtual machine doesn't incur charges.

WARNING The difference between *stopping* and *terminating* a virtual machine is important. You can start a stopped virtual machine. This isn't possible with a terminated virtual machine. If you terminate a virtual machine, you delete it.

Figure 3.17 illustrates the difference between stopping and terminating an EC2 instance, with the help of a flowchart.

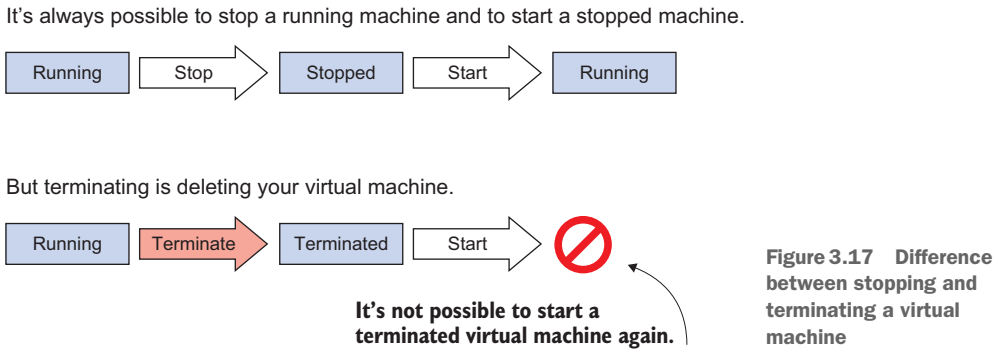


Figure 3.17 Difference between stopping and terminating a virtual machine

Stopping or terminating unused virtual machines saves costs and prevents you from being surprised by an unexpected bill from AWS. You may want to stop or terminate unused virtual machines when:

- *You have launched virtual machines to implement a proof-of-concept.* After finishing the project, the virtual machines are no longer needed. Therefore, you can terminate them.
- *You are using a virtual machine to test a web application.* As no one else uses the virtual machine, you can stop it before you knock off work, and start it back up again the following day.
- *One of your customers canceled their contract.* After backing up relevant data, you can terminate the virtual machines that had been used for your former customer.

After you terminate a virtual machine, it's no longer available and eventually disappears from the list of virtual machines.

Cleaning up

Terminate the virtual machine named `mymachine` that you started at the beginning of this chapter:

1. Open the EC2 service from the main navigation, and select Instances from the submenu.
2. Select the running virtual machine by clicking the row in the table.
3. In the Actions menu, choose Instance State > Terminate.

3.4 Changing the size of a virtual machine

It is always possible to change the size of a virtual machine. This is one of the advantages of using the cloud, and it gives you the ability to scale vertically. If you need more computing power, increase the size of the EC2 instance.

In this section, you'll learn how to change the size of a running virtual machine. To begin, follow these steps to start a small virtual machine:

1. Open the AWS Management Console, and choose the EC2 service.
2. Start the wizard to launch a new virtual machine by clicking Launch Instance.
3. Select Ubuntu Server 16.04 LTS (HVM) as the AMI for your virtual machine.
4. Choose the instance type t2.micro.
5. Click Review and Launch to start the virtual machine.
6. Click Edit Security Groups to configure the firewall. Choose Select an Existing Security Group and select the security group named `ssh-only`.
7. Click Review and Launch to start the virtual machine.
8. Check the summary for the new virtual machine, and click Launch.

9. Choose the option Choose an Existing Key Pair, select the key pair `mykey`, and click Launch Instances.
10. Switch to the overview of EC2 instances, and wait for the new virtual machine's state to switch to Running.

You've now started an EC2 instance of type t2.micro. This is one of the smallest virtual machines available on AWS.

Use SSH to connect to your virtual machine, as shown in the previous section, and execute `cat /proc/cpuinfo` and `free -m` to see information about the machine's capabilities. The output should look similar to this:

```
$ cat /proc/cpuinfo
processor   : 0
vendor_id   : GenuineIntel
cpu family  : 6
model       : 63
model name  : Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz
stepping    : 2
microcode   : 0x36
cpu MHz     : 2400.054
cache size  : 30720 KB
[...]

$ free -m
              total        used        free      shared  buff/cache   available
Mem:            990          42         632           3         315         794
Swap:             0           0           0
```

Your virtual machine provides a single CPU core and 990 MB of memory. If your application is having performance issues, increasing the instance size can solve the problem. Use your machine's metrics as described in section 3.2 to find out if you are running out of CPU or networking capacity. Would your application benefit from additional memory? If so, increasing the instance size will improve the application's performance as well.

If you need more CPUs, more memory, or more networking capacity, there are many other sizes to choose from. You can even change the virtual machine's instance family and generation. To increase the size of your VM, you first need to stop it:

1. Open the AWS Management Console, and choose the EC2 service.
2. Click Instances in the submenu to jump to an overview of your virtual machines.
3. Select your running VM from the list by clicking it.
4. Choose Stop from the Actions menu.

WARNING Starting a virtual machine with instance type `m4.large` incurs charges. Go to http://aws.amazon.com/ec2/pricing if you want to see the current on-demand hourly price for an m4.large virtual machine.

After waiting for the virtual machine to stop, you can change the instance type:

1. Choose Change Instance Type from the Actions menu under Instance Settings. As shown in figure 3.18, a dialog opens in which you can choose the new instance type for your VM.
2. Select m4.large for Instance Type.
3. Save your changes by clicking Apply.

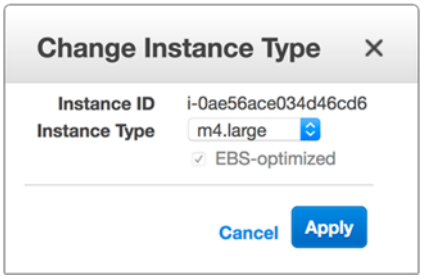


Figure 3.18 Increasing the size of your virtual machine by selecting m4.large as Instance Type

You've now changed the size of your virtual machine and are ready to start it again. To do so, select your virtual machine and choose Start from the Actions menu under Instance State. Your VM will start with more CPUs, more memory, and more networking capabilities. The public and private IP addresses have also changed. Grab the new public DNS to reconnect via SSH; you'll find it in the VM's Details view.

Use SSH to connect to your EC2 instance, and execute `cat /proc/cpuinfo` and `free -m` to see information about its CPU and memory. The output should look similar to this:

```
$ cat /proc/cpuinfo
processor    : 0
vendor_id    : GenuineIntel
cpu family   : 6
model        : 79
model name   : Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz
stepping     : 1
microcode    : 0xb00001d
cpu MHz      : 2300.044
cache size   : 46080 KB
[...]

processor    : 1
vendor_id    : GenuineIntel
cpu family   : 6
model        : 79
model name   : Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz
stepping     : 1
microcode    : 0xb00001d
cpu MHz      : 2300.044
cache size   : 46080 KB
[...]

$ free -m
              total        used        free      shared  buff/cache   available
Mem:           7982          62        7770           8         149        7701
Swap:             0           0           0
```

Your virtual machine can use two CPU cores and offers 7,982 MB of memory, compared to a single CPU core and 990 MB of memory before you increased the VM's size.

Cleaning up

Terminate the m4.large VM to stop paying for it:

1. Open the EC2 service from the main navigation, and select Instances from the submenu.
2. Select the running virtual machine by clicking the row in the table.
3. In the Actions menu, choose Instance State > Terminate.

3.5 Starting a virtual machine in another data center

AWS offers data centers all over the world. Take the following criteria into account when deciding which region to choose for your cloud infrastructure:

- *Latency*—Which region offers the shortest distance between your users and your infrastructure?
- *Compliance*—Are you allowed to store and process data in that country?
- *Service availability*—AWS does not offer all services in all regions. Are the services you are planning to use available in the region? Check out the service availability region table at http://mng.bz/0q40.
- *Costs*—Service costs vary by region. Which region is the most cost-effective region for your infrastructure?

Let's assume you have customers not just in the United States but in Australia as well. At the moment you are only operating EC2 instances in N. Virginia (US). Customers from Australia complain about long loading times when accessing your website. To make your Australian customers happy, you decide to launch an additional VM in Australia.

Changing a data center is simple. The Management Console always shows the current data center you're working in, on the right side of the main navigation menu. So far, you've worked in the data center N. Virginia (US), called us-east-1. To change the data center, click N. Virginia and select Sydney from the menu. Figure 3.19 shows how to jump to the data center in Sydney called ap-southeast-2.

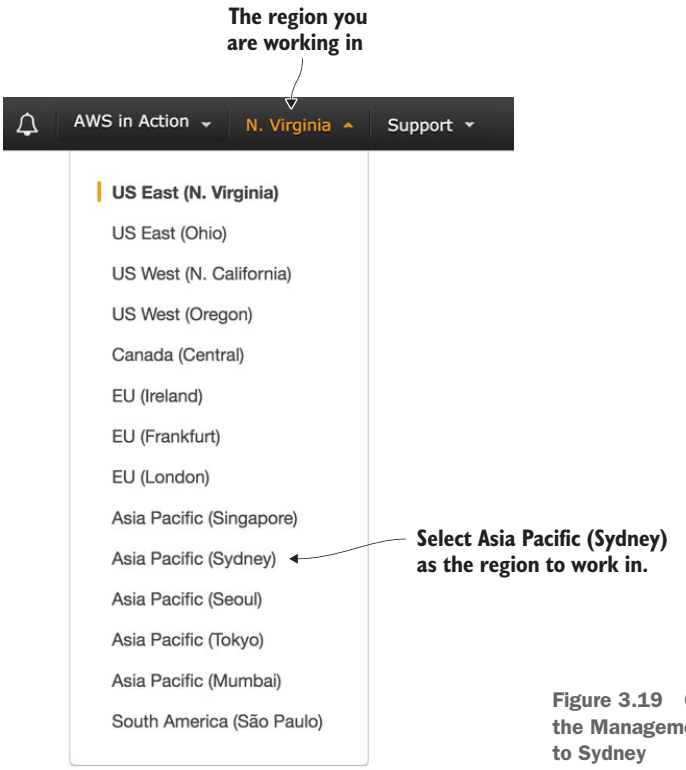


Figure 3.19 Changing the data center in the Management Console from N. Virginia to Sydney

AWS groups its data centers into these regions:

- US East, N. Virginia (us-east-1)
- US West, N. California (us-west-1)
- Canada, Central (ca-central-1)
- EU, Frankfurt (eu-central-1)
- EU, Paris (eu-west-3)
- Asia Pacific, Seoul (ap-northeast-2)
- Asia Pacific, Sydney (ap-southeast-2)
- South America, São Paulo (sa-east-1)
- US East, Ohio (us-east-2)
- US West, Oregon (us-west-2)
- EU, Ireland (eu-west-1)
- EU, London (eu-west-2)
- Asia Pacific, Tokyo (ap-northeast-1)
- Asia Pacific, Singapore (ap-southeast-1)
- Asia Pacific, Mumbai (ap-south-1)

You can specify the region for most AWS services. The regions are independent of each other; data isn't transferred between regions. Typically, a region is a collection of three or more data centers located in the same area. Those data centers are well connected to each other and offer the ability to build a highly available infrastructure, as you'll discover later in this book. Some AWS services, like CDN and the Domain Name System (DNS) service, act globally on top of these regions and even on top of some additional data centers.

After you change to the EC2 service in the Management Console, you may wonder why no key pair is listed in the EC2 overview. You created a key pair for SSH logins in the region N. Virginia (US). But the regions are independent, so you have to create a new key pair for the Sydney region. Follow these steps (see section 1.2 if you need more details):

1. Open the EC2 service from the main navigation, and select Key Pairs from the submenu.
2. Click Create Key Pair, and type in `sydney` as the key pair name.
3. Download and save the key pair.
4. Windows only: Open PuTTYgen, and select SSH-2 RSA under Type of Key to Generate. Click Load. Select the sydney.pem file, and click Open. Confirm the dialog box. Click Save Private Key.
5. Linux and macOS only: Change the access rights of the file sydney.pem by running `chmod 400 sydney.pem` in the terminal.

You're ready to start a virtual machine in the data center in Sydney. Follow these steps to do so:

1. Open the EC2 service from the main navigation menu, and select Instances from the submenu.
2. Click Launch Instance to start a wizard that will guide you through starting a new virtual machine.
3. Select the Amazon Linux AMI (HVM) machine image.
4. Choose t2.micro as the instance type, and click Review and Launch to go to the next screen.
5. Click Edit Security Groups to configure the firewall. Change the Security Group Name to `webserver` and the Description to `HTTP and SSH`. Add two rules: one of type SSH and another of type HTTP. Allow access to SSH and HTTP from anywhere by defining `0.0.0.0/0` as the source for both rules. Your firewall configuration should look like figure 3.20. Click Review and Launch.
6. Click Launch, and select `sydney` as the existing key pair to use for launching your VM.
7. Click View Instances to see the overview of virtual machines, and wait for your new VM to start.

You're finished! A virtual machine is running in a data center in Sydney. Let's proceed with installing a web server on it. To do so, you have to connect to your virtual machine via SSH. Grab the current public IP address of your virtual machine from its Details page.

Open a terminal, and type `ssh -i` *`$PathToKey`*`/sydney.pem ec2-user@$PublicIp`, replacing *`$PathToKey`* with the path to the key file sydney.pem that you downloaded, and *`$PublicIp`* with the public IP address from the details of your virtual machine. Answer Yes to the security alert about the authenticity of the new host.

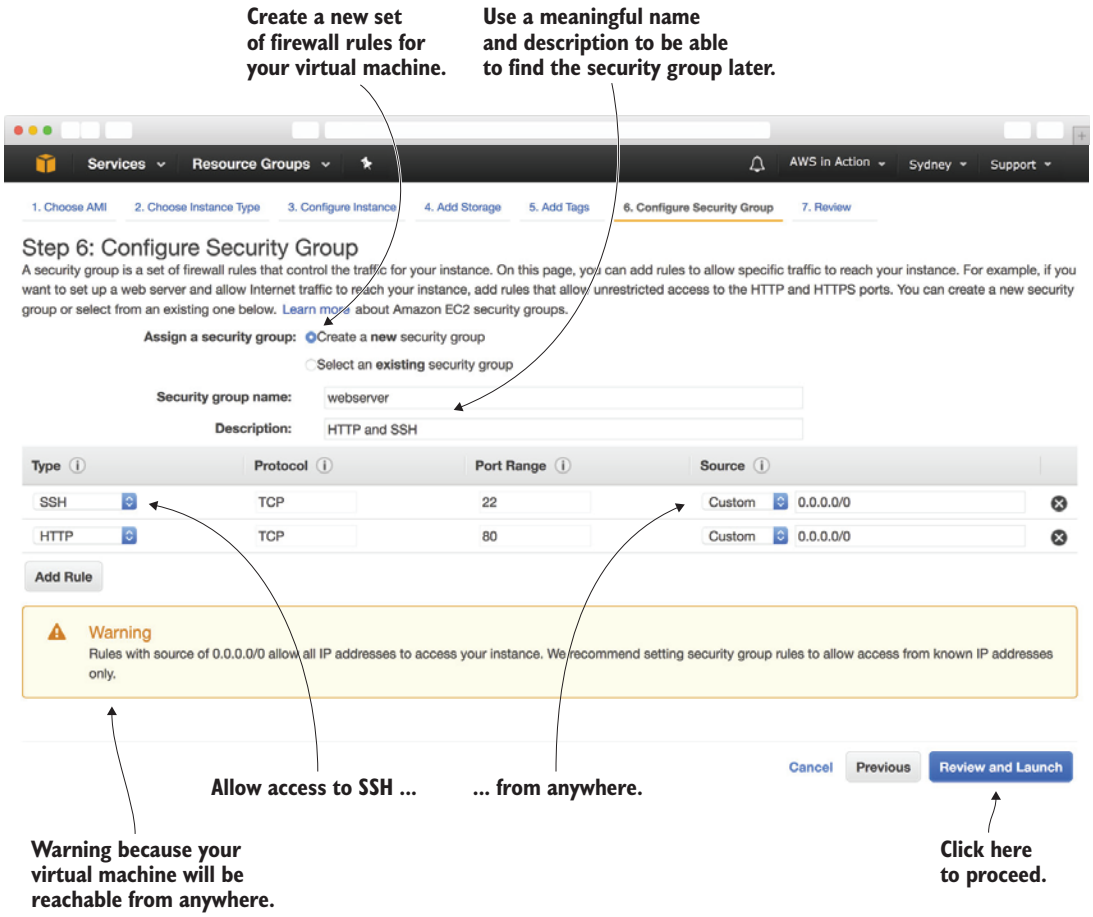


Figure 3.20 Configuring the firewall for a web server in Sydney

Windows

Find the sydney.ppk file you created after downloading the new key pair, and open it by double-clicking. The PuTTY Pageant should appear in the taskbar as an icon. Next, start PuTTY and connect to the public IP address you got from the Details page of your virtual machine. Answer Yes to the security alert regarding the authenticity of the new host, and type in `ec2-user` as login name. Press Enter.

To serve your website to your Australian customers, connect to the EC2 instance via SSH and install a web server by executing `sudo yum install httpd -y`. To start the web server, type `sudo service httpd start` and press Return to execute the command. Your web browser should show a placeholder site if you open http://*$PublicIp*, with *$PublicIp* replaced by the public IP address of your virtual machine.

NOTE You're using two different operating systems in this chapter. You started with a VM based on Ubuntu at the beginning of the chapter. Now you're using Amazon Linux, a distribution based on Red Hat Enterprise Linux. That's why you have to execute different commands to install software. Ubuntu uses `apt-get` and Amazon Linux is using `yum`.

Next, you'll attach a fixed public IP address to the virtual machine.

3.6 Allocating a public IP address

You've already launched some virtual machines while reading this book. Each VM was connected to a public IP address automatically. But every time you launched or stopped a VM, the public IP address changed. If you want to host an application under a fixed IP address, this won't work. AWS offers a service called *Elastic IPs* for allocating fixed public IP addresses. You can allocate a public IP address and associate it with an EC2 instance by following these steps in figure 3.21:

1. Open the Management Console, and go to the EC2 service.
2. Choose Elastic IPs from the submenu. You'll see an overview of public IP addresses.
3. Allocate a public IP address by clicking Allocate New Address.
4. Confirm by clicking on Allocate.
5. Your fixed public IP address is shown. Click Close to go back to the overview.

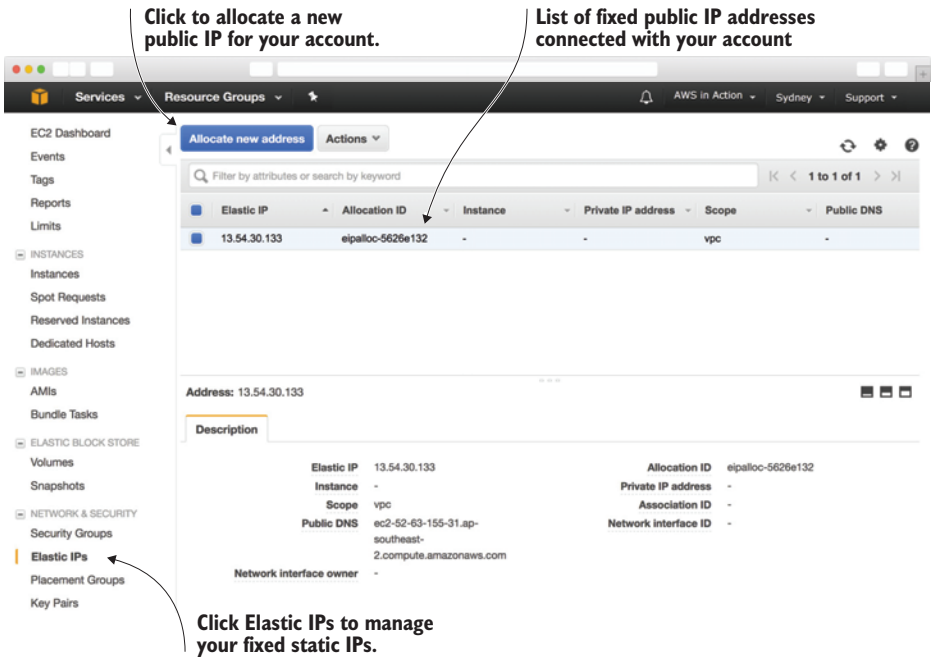


Figure 3.21 Overview of public IP addresses connected to your account in the current region

Now you can associate the public IP address with a virtual machine of your choice:

1. Select your public IP address, and choose Associate Address from the Actions menu. A dialog similar to figure 3.22 appears.
2. Select Instance as the Resource Type.
3. Enter your EC2 instance's ID in the Instance field. There is only a single virtual machine running at the moment, so only one option is available.
4. Only one Private IP is available for your virtual machine. Select it.
5. Click Associate to finish the process.

Your virtual machine is now accessible through the public IP address you allocated at the beginning of this section. Point your browser to this IP address, and you should see the placeholder page as you did in section 3.5.

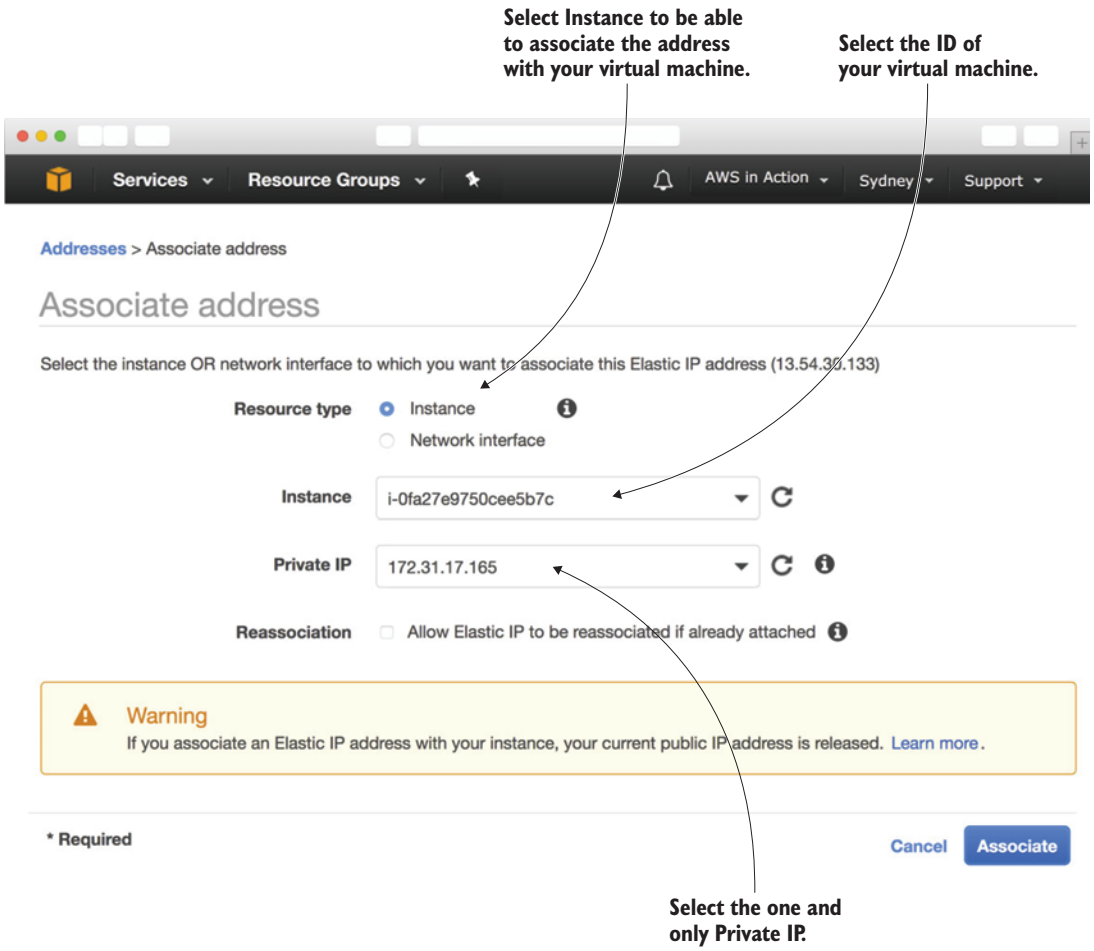


Figure 3.22 Associating a public IP address with your EC2 instance

Allocating a public IP address can be useful if you want to make sure the endpoint to your application doesn't change, even if you have to replace the virtual machine behind the scenes. For example, assume that virtual machine A is running and has an associated Elastic IP. The following steps let you replace the virtual machine with a new one without changing the public IP address:

1. Start a new virtual machine B to replace running virtual machine A.
2. Install and start applications as well as all dependencies on virtual machine B.
3. Disassociate the Elastic IP from virtual machine A, and associate it with virtual machine B.

Requests using the Elastic IP address will now be routed to virtual machine B, with a short interruption while moving the Elastic IP. You can also connect multiple public IP addresses with a virtual machine by using multiple network interfaces, as described in the next section. This can be useful if you need to host different applications running on the same port, or if you want to use a unique fixed public IP address for different websites.

> **WARNING** IPv4 addresses are scarce. To prevent stockpiling Elastic IP addresses, AWS will charge you for Elastic IP addresses that aren't associated with a virtual machine. You'll clean up the allocated IP address at the end of the next section.

3.7 Adding an additional network interface to a virtual machine

In addition to managing public IP addresses, you can control your virtual machine's network interfaces. It is possible to add multiple network interfaces to a VM and control the private and public IP addresses associated with those network interfaces.

Here are some typical use cases for EC2 instances with multiple network interfaces:

- Your web server needs to answer requests by using multiple TLS/SSL certificates, and you can't use the Server Name Indication (SNI) extension due to legacy clients.
- You want to create a management network separated from the application network, and therefore your EC2 instance needs to be accessible from two networks. Figure 3.23 illustrates an example.
- Your application requires or recommends the use of multiple network interfaces (for example, network and security appliances).

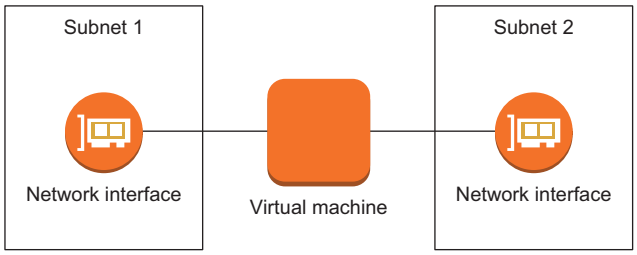


Figure 3.23 A virtual machine with two network interfaces in two different subnets

You use an additional network interface to connect a second public IP address to your EC2 instance. Follow these steps to create an additional networking interface for your virtual machine:

1. Open the Management Console, and go to the EC2 service.
2. Select Network Interfaces from the submenu.
3. The default network interface of your virtual machine is shown in the list. Note the subnet ID of the network interface.
4. Click Create Network Interface. A dialog like the one shown in figure 3.24 appears.
5. Enter `2nd interface` as the description.
6. Choose the subnet you noted down in step 3.
7. Leave Private IP Address empty. A private IP will be assigned to the network interface automatically.
8. Select the Security Groups that have *webserver* in their description.
9. Click Yes, Create.

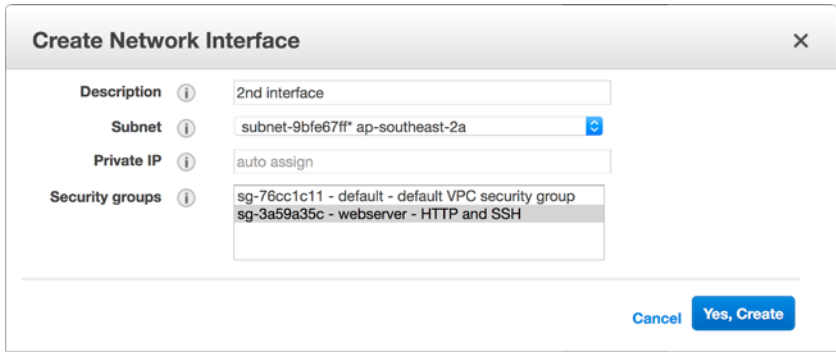


Figure 3.24 Creating an additional networking interface for your virtual machine

After the new network interface's state changes to Available, you can attach it to your virtual machine. Select the new 2nd interface network interface, and choose Attach from the menu. A dialog opens like shown in figure 3.25. Choose the only available Instance ID, and click Attach.

You've attached an additional networking interface to your virtual machine. Next, you'll connect an additional public IP address to the additional networking interface. To do so, note down the network interface ID of the 2nd interface shown in the overview—`eni-b847f4c5` in our example—and follow these steps:

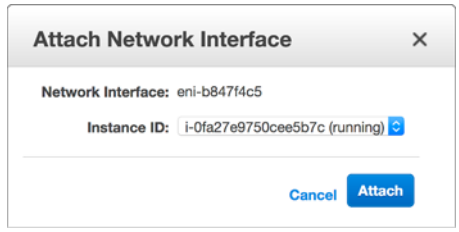


Figure 3.25 Attaching an additional networking interface to your virtual machine

1. Open the AWS Management Console, and go to the EC2 service.
2. Choose Elastic IPs from the submenu.
3. Click Allocate New Address to allocate a new public IP address, as you did in section 3.6.
4. Select your public IP address, and choose Associate Address from the Actions menu. A dialog similar to figure 3.26 appears.
5. Select Network interface as the Resource Type.
6. Enter your 2nd interface's ID in the Network Interface field.
7. Select the only available Private IP for your network interface.
8. Click Associate to finish the process.

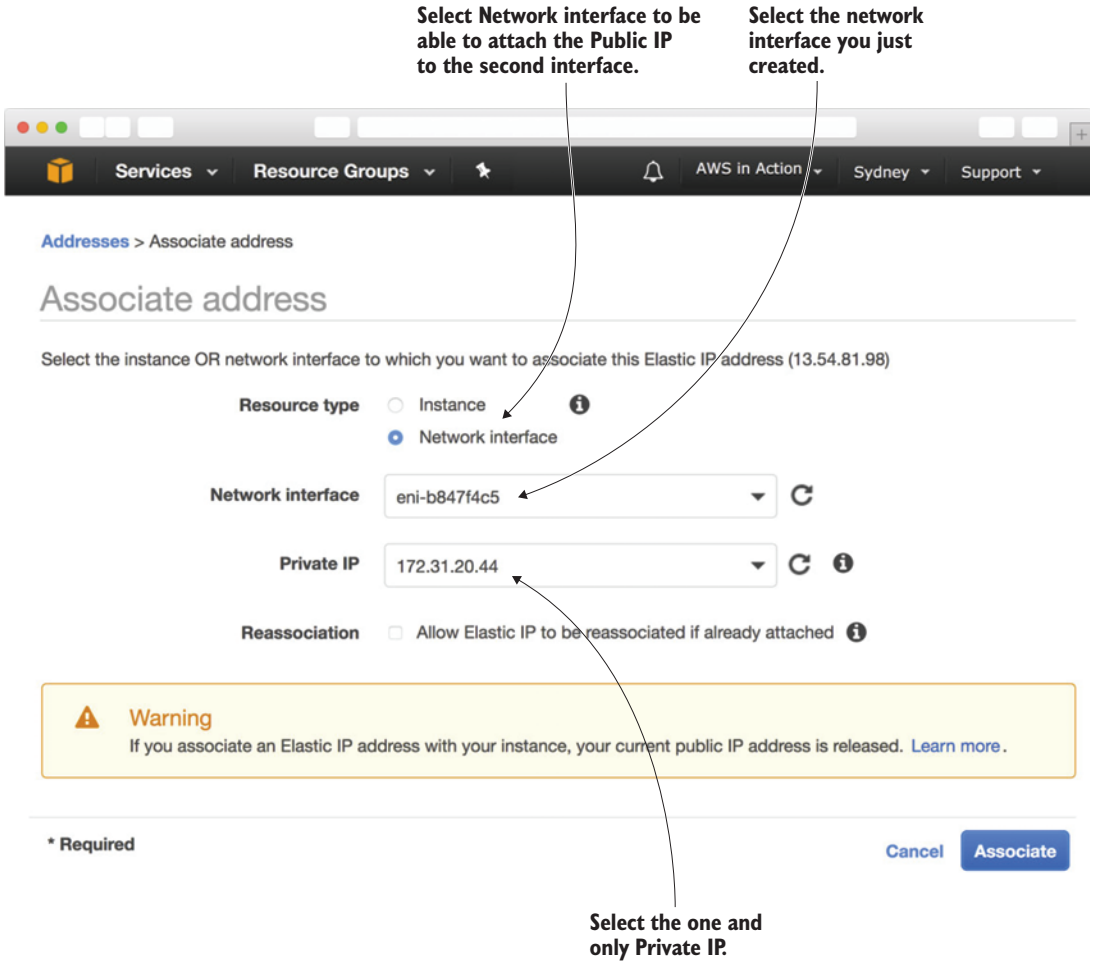


Figure 3.26 Associating a public IP address with the additional networking interface

Your virtual machine is now reachable under two different public IP addresses. This enables you to serve two different websites, depending on the public IP address. You need to configure the web server to answer requests depending on the public IP address.

If you connect to your virtual machine via SSH and insert `ifconfig` into the terminal, you can see your new networking interface attached to the virtual machine, as shown in the following output after running the `ifconfig` command:

```
$ ifconfig
eth0      Link encap:Ethernet  HWaddr 02:06:EE:59:F7:65
          inet addr:172.31.17.165  Bcast:172.31.31.255  Mask:255.255.240.0
          inet6 addr: fe80::6:eeff:fe59:f765/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9001  Metric:1
          RX packets:3973 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2648 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5104781 (4.8 MiB)  TX bytes:167733 (163.8 KiB)

eth1      Link encap:Ethernet  HWaddr 02:03:FA:95:9B:BB
          inet addr:172.31.20.44  Bcast:172.31.31.255  Mask:255.255.240.0
          inet6 addr: fe80::3:faff:fe95:9bbb/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9001  Metric:1
          RX packets:84 errors:0 dropped:0 overruns:0 frame:0
          TX packets:87 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9997 (9.7 KiB)  TX bytes:11957 (11.6 KiB)
[...]
```

Each network interface is connected to a private and a public IP address. You'll need to configure the web server to deliver different websites depending on the IP address. Your virtual machine doesn't know anything about its public IP address, but you can distinguish the requests based on the private IP address.

First you need two websites. Run the following commands via SSH on your virtual machine in Sydney to download two simple placeholder websites:

```
$ sudo -s
$ mkdir /var/www/html/a
$ wget -P /var/www/html/a https://raw.githubusercontent.com/AWSinAction/\
➥ code2/master/chapter03/a/index.html
$ mkdir /var/www/html/b
$ wget -P /var/www/html/b https://raw.githubusercontent.com/AWSinAction/\
➥ code2/master/chapter03/b/index.html
```

Next you need to configure the web server to deliver the websites depending on which IP address is called. To do so, add a file named `a.conf` under /etc/httpd/conf.d with the following content. Change the IP address from 172.31.x.x to the IP address from the `ifconfig` output for the networking interface eth0:

```
<VirtualHost 172.31.x.x:80>
  DocumentRoot /var/www/html/a
</VirtualHost>
```

Repeat the same process for a configuration file named `b.conf` under /etc/httpd/conf.d with the following content. Change the IP address from 172.31.y.y to the IP address from the `ifconfig` output for the networking interface eth1:

```
<VirtualHost 172.31.y.y:80>
  DocumentRoot /var/www/html/b
</VirtualHost>
```

To activate the new web server configuration, execute `sudo service httpd restart` via SSH. Change to the Elastic IP overview in the Management Console. Copy both public IP addresses, and open them with your web browser. You should get the answer "Hello A!" or "Hello B!" depending on the public IP address you're calling. Thus you can deliver two different websites, depending on which public IP address the user is calling. Congrats—you're finished!

Cleaning up

It's time to clean up your setup:

1. Terminate the virtual machine.
2. Go to Networking Interfaces, and select and delete the networking interface.
3. Change to Elastic IPs, and select and release the two public IP addresses by clicking Release Addresses from the Actions menu.
4. Go to Key Pairs, and delete the sydney key pair you created.
5. Go to Security Groups, and delete the webserver security group you created.

That's it. Everything is cleaned up, and you're ready for the next section.

NOTE You switched to the AWS region in Sydney earlier. Now you need to switch back to the region US East (N. Virginia). You can do so by selecting US East (N. Virginia) from the region chooser in the main navigation menu of the Management Console.

3.8 Optimizing costs for virtual machines

Usually you launch virtual machines *on demand* in the cloud to gain maximum flexibility. AWS calls them on-demand instances, because you can start and stop VMs on-demand, whenever you like, and you're billed for every second or hour the machine is running.

There are two options to reduce EC2 costs: *spot instances* or *reserved instances*. Both help to reduce costs but decrease your flexibility. With a spot instance, you bid for unused capacity in an AWS data center; the price is based on supply and demand. You can use reserved instances if you need a virtual machine for a year or longer; you agree to pay for the given time frame and receive a discount in advance. Table 3.2 shows the differences between these options.

Billing unit: Seconds

Most EC2 instances running Linux (such as Amazon Linux or Ubuntu) are billed per second. The minimum charge per instance is 60 seconds. For example, if you terminate a newly launched instance after 30 seconds, you have to pay for 60 seconds. But if you terminate an instance after 61 seconds, you pay exactly for 61 seconds.

An EC2 instance running Microsoft Windows or a Linux distribution with an extra hourly charge (such as Red Hat Enterprise Linux or SUSE Linux Enterprise Server) is not billed per second, but per hour. A minimum charge of one hour applies. The same is true for EC2 instances with an extra hourly charge launched from the AWS Marketplace.

Table 3.2 Differences between on-demand, reserved, and spot virtual machines

	On-demand	Reserved	Spot
Price	High	Medium	Low
Flexibility	High	Low	Medium
Reliability	Medium	High	Low
	Dynamic workloads (for example, for a news site) or proof-of-concept	Predictable and static workloads (for example, for a business application)	Batch workloads (for example, for data analytics, media encoding, ...)

3.8.1 Reserve virtual machines

Reserving a virtual machine means committing to using a specific VM in a specific data center. You have to pay for a reserved VM whether or not it's running. In return, you benefit from a price reduction of up to 60%. On AWS, you can choose one of the following options if you want to reserve a virtual machine:

- No Upfront, 1-year term
- Partial Upfront, 1- or 3-year term
- All Upfront, 1- or 3-year term

Table 3.3 shows what this means for an m4.large virtual machine with two virtual CPUs and 8 GB of memory.

WARNING Buying a reservation will incur costs for 1 or 3 years. That's why we did not add an example for this section.

Table 3.3 Potential cost savings for a virtual machine with instance type m4.large

	Monthly cost	Upfront cost	Effective monthly cost	Savings vs. on-demand
On-demand	$73.20 USD	$0.00 USD	$73.20 USD	N/A
No Upfront, 1-year term, standard	$45.38 USD	$0.00 USD	$45.38 USD	38%

Table 3.3 Potential cost savings for a virtual machine with instance type m4.large *(continued)*

	Monthly cost	Upfront cost	Effective monthly cost	Savings vs. on-demand
Partial Upfront, 1-year term, standard	$21.96 USD	$258.00 USD	$43.46 USD	40%
All Upfront, 1-year term, standard	$0.00 USD	$507.00 USD	$42.25 USD	42%
No Upfront, 3-year term, standard	$31.47 USD	$0.00 USD	$31.47 USD	57%
Partial Upfront, 3-year term, standard	$14.64 USD	$526.00 USD	$29.25 USD	60%
All Upfront, 3-year term, standard	$0.00 USD	$988.00 USD	$27.44 USD	63%

You can trade cost reductions against flexibility by reserving virtual machines on AWS. There are different options offering different levels of flexibility when buying a reserved instance:

- Reserved instances, either with or without capacity reservation
- Standard or convertible reserved instances

We'll explain these in more detail in the following sections. It is possible to buy reservations for specific time frames, called Scheduled Reserved Instances. For example, you can make a reservation for every workday from 9 a.m. to 5 p.m.

RESERVED INSTANCES WITH CAPACITY RESERVATION

If you own a reservation for a virtual machine (a *reserved instance*) in a specific availability zone, the capacity for this virtual machine is reserved for you in the public cloud. Why is this important? Suppose demand increases for VMs in a particular data center, perhaps because another data center broke down and many AWS customers have to launch new virtual machines to replace their broken ones. In this rare case, the orders for on-demand VMs will pile up, and it may become nearly impossible to start a new VM in that data center. If you plan to build a highly available setup across multiple data centers, you should also think about reserving the minimum capacity you'll need to keep your applications running. The downside of a capacity reservation is low flexibility. You commit to pay for a VM of a specific type within a specific data center for one or three years.

RESERVED INSTANCES WITHOUT CAPACITY RESERVATION

If you choose not to reserve capacity for your virtual machine, you can profit from reduced hourly prices and more flexibility. A reserved instance without capacity reservation is valid for VMs in a whole region, no matter which specific data center is used. The reservation is applicable to all instance sizes of an instance family as well. For example, if you buy a reservation for a m4.xlarge machine, you can also run two m4.large machines making use of the reservation as well.

Modifying reservations

It is possible to modify a reservation without additional charges. Doing so allows you to adapt your reservations to changes in your workload over time. You can:

- Toggle capacity reservation.
- Modify which data center your reservation is in.
- Split or merge a reservation. For example, you can merge two t2.small reservations into a t2.medium reservation.

STANDARD OR CONVERTIBLE?

When buying a reservation for three years, you have the option to chose between standard or convertible offerings. A *standard reservation* limits you to a specific instance family (such as m4). A *convertible reservation* can be exchanged for another reservation, for example to change to a different instance family. Convertible reservations are more expensive than standard reservations but offer higher flexibility.

Being able to switch to another instance family might be valuable because AWS might introduce a new instance family with lower prices and higher resources in the future. Or perhaps your workload pattern changes and you want to switch the instance family, for example from a general-purpose family to a one that's optimized for computing.

We recommend that you start with on-demand machines and switch to a mix of on-demand and reserved instances later. Depending on your high availability requirements, you can choose between reservations with and without capacity reservation.

3.8.2 Bidding on unused virtual machines

In addition to reserved virtual machines, there is another option for reducing costs: *spot instances*. With a spot instance, you bid for unused capacity in the AWS cloud. A *spot market* is a market where standardized products are traded for immediate delivery. The price of the products on the market depends on supply and demand. On the AWS spot market, the products being traded are VMs, and they're delivered by starting a virtual machine.

Figure 3.27 shows the price chart for a specific instance type for a virtual machine. If the current spot price is lower than your maximum bid for a specific virtual machine VM in a specific data center, your spot request will be fulfilled, and a virtual machine will start, billed at the spot price. If the current spot price exceeds your bid, your VM will be terminated (not stopped) by AWS after two minutes.

The spot price can be more or less flexible depending on the size of the VMs and the data center. We've seen everything from a spot price that was only 10% of the on-demand price to a spot price that was greater than the on-demand price. As soon as the spot price exceeds your bid, your EC2 instance will be terminated within two minutes. You shouldn't use spot instances for tasks like web or mail servers, but you can use them to run asynchronous tasks like analyzing data or encoding media assets. You

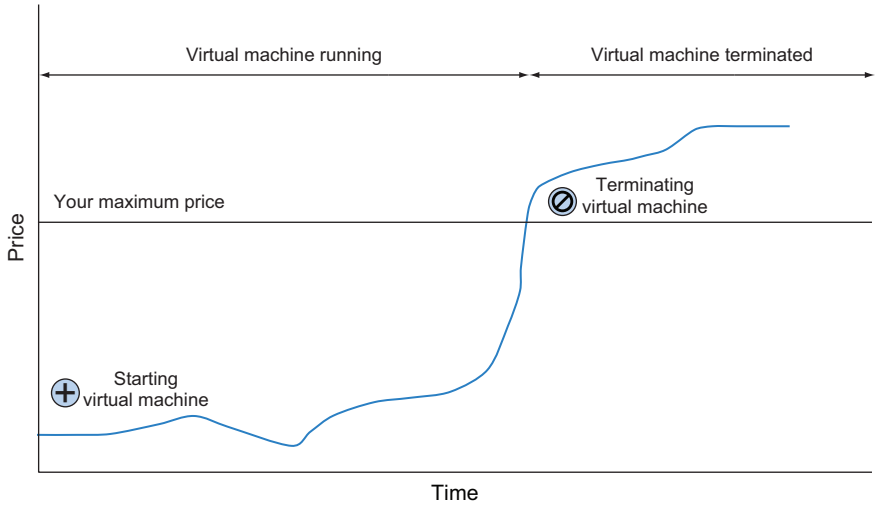


Figure 3.27 Functionality of the spot market for virtual machines

can even use a spot instance to check for broken links on your website, as you did in section 3.1, because this isn't a time-critical task.

Let's start a new virtual machine that uses the price reductions of the spot market. First you have to place your order on the spot market; figure 3.28 shows the starting point for requesting a spot instance. You get there by choosing the EC2 service from the main navigation menu and selecting Spot Requests from the submenu. Click on Pricing History. A dialog appears showing the spot prices for virtual machines; historical prices are available for the different server sizes and different data centers.

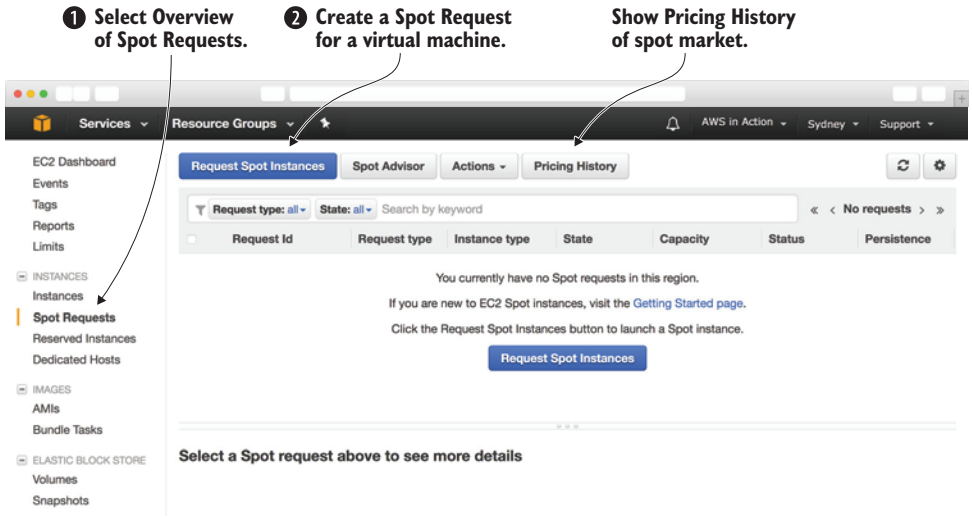


Figure 3.28 Requesting a spot instance

Click Request Spot Instances to start the wizard that guides you through the process of requesting a spot instance.

> **WARNING** Starting a virtual machine with instance type m3.medium via spot request incurs charges. The maximum price (bid) is $0.093 in the following example.

The steps in figure 3.29 are necessary to start the smallest available virtual machine available on the spot market:

1. You can choose between three request types. *Request* will request a one-time virtual machine. *Request and Maintain* will create a fleet of VMs and keep them running at the lowest possible price. Or *Reserve for Duration* will request a virtual machine with a guaranteed lifetime of 1 to 6 hours. Choose *Request* to request a single virtual machine.
2. Request a target capacity of one EC2 instance.
3. Select the Ubuntu 16.04 LTS AMI.
4. Choose m3.medium, the smallest available instance type for spot instances.
5. Keep the defaults for allocation strategy and network.
6. Set your own maximum price. In this example, we're using the on-demand price of $0.093 USD as a maximum. Prices are subject to change. Get the latest prices from https://aws.amazon.com/ec2/pricing/on-demand/, the Amazon EC2 pricing page.
7. Click the Next button to proceed with the next step.

The next step, as shown in figure 3.30, is to configure the details of your virtual machine and the spot request. Set the following parameters:

1. Keep the default storage settings. A network-attached volume with 8 GB will be enough to run the link checker.
2. Keep the defaults for the EC2 instance as well, expect for the Instance tags. Add a tag with key `Name` and value `spotinstance` so that you can identify your virtual machine later.
3. Select your Key Pair named `mykey` and keep the default for IAM instance profile and IAM fleet role.
4. Select the `ssh-only` security group, which you created at the beginning of the chapter to allow incoming SSH traffic to your virtual machine.
5. Keep the default for the Public IP as well as the request validity.
6. Click the Review button to proceed to the next step.

The last step of the wizard shows a summary of your spot request. Click the Launch button to create the request.

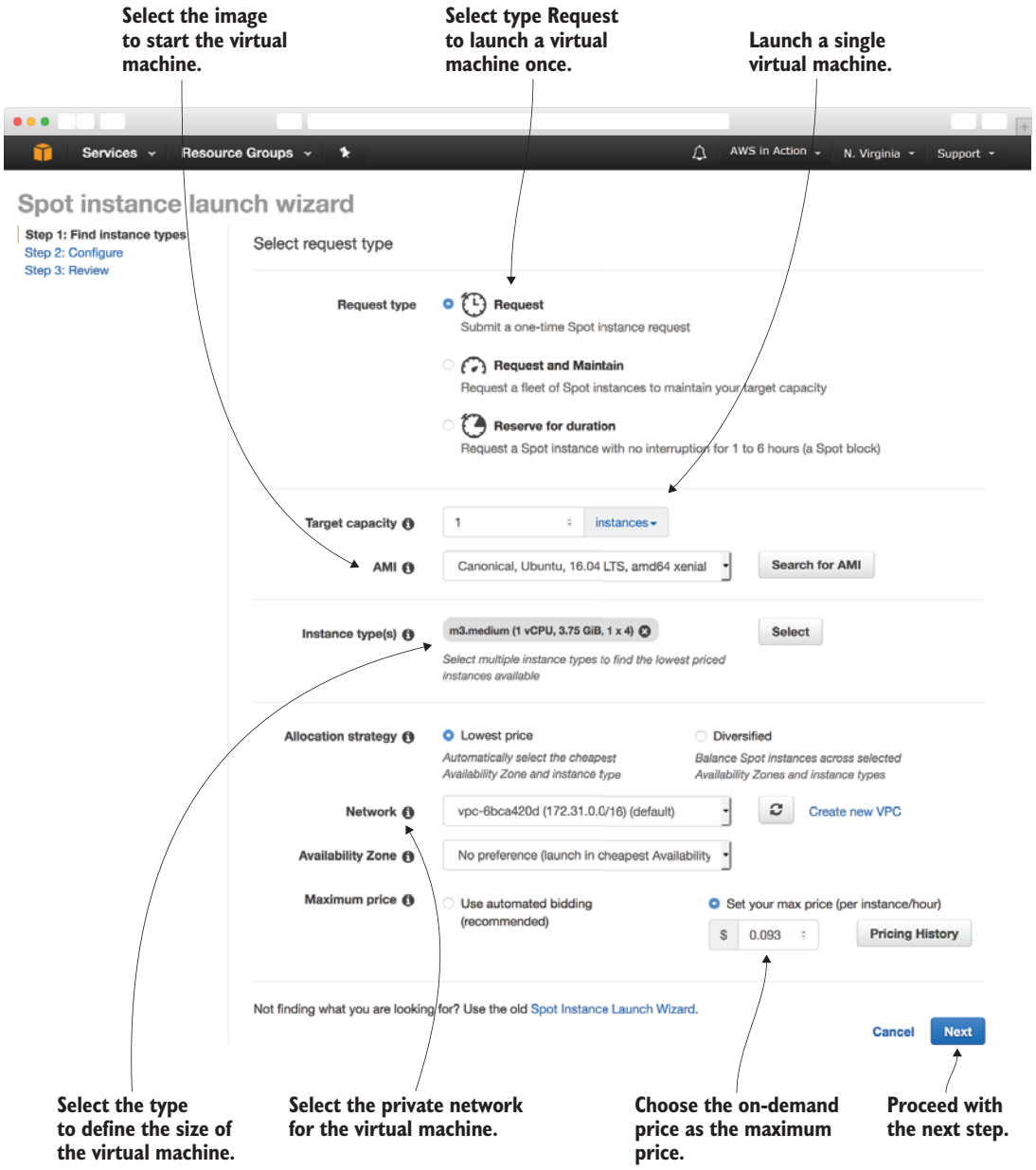


Figure 3.29 Requesting a spot instance (step 1)

Figure 3.30 Requesting a spot instance (step 2)

After you finish the wizard, your request for a virtual machine is placed on the market. Your spot request appears within the overview shown in figure 3.31. It may take several minutes for your request to be fulfilled. Look at the Status field for your request: because the spot market is unpredictable, it's possible that your request could fail. If this happens, repeat the process to create another request, and choose another instance type.

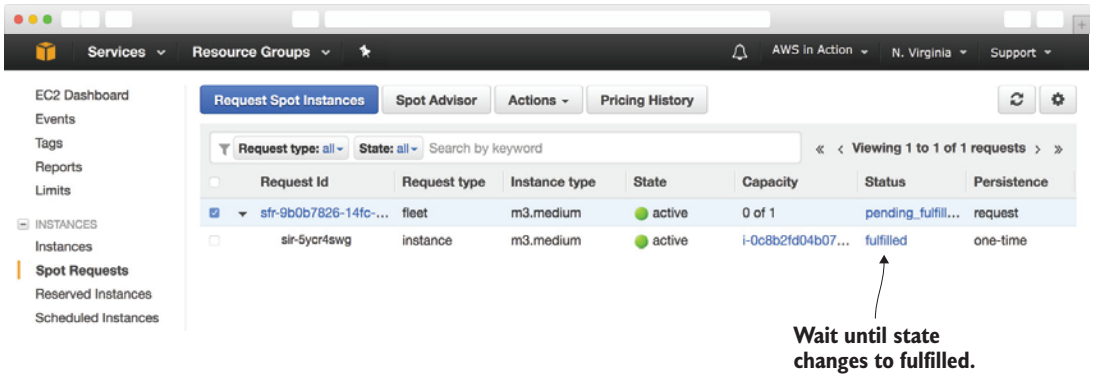


Figure 3.31 Waiting for the spot request to be fulfilled and the virtual machine to start

When the state of your requests flips to fulfilled, a virtual machine is started. You can look at it after switching to Instances via the submenu; you'll find a running or starting instance listed in the overview of virtual machines. You've successfully started a virtual machine that is billed as spot instance! You can now use SSH to connect to the VM and run the link checker as described in section 3.1.3.

Cleaning up

Terminate the m3.medium VM to stop paying for it:

1. Open the EC2 service from the main navigation menu, and select Spot Requests from the submenu.
2. Select your spot request by clicking the first row in the table.
3. In the Actions menu, choose Cancel Spot request.
4. Make sure the option Terminate Instances is selected, click OK to confirm.

Summary

- You can choose an OS when starting a virtual machine.
- Using logs and metrics can help you to monitor and debug your VM.
- If you want to change the size of your VM, you can change the number of CPUs, as well as the amount of memory and storage.

- You can start VMs in different regions all over the world to provide low latency for your users.
- Allocating and associating a public IP address to your virtual machine gives you the flexibility to replace a VM without changing the public IP address.
- You can save on costs by reserving virtual machines or bidding for unused capacity on the virtual machine spot market.